SF 298 MASTER COPY KEEP THIS COPY FOR REPRODUCTION PURPOSES

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 24th 1998	Final report 9-28-95 to 9-27-98

4. TITLE AND SUBTITLE
Methods for Intelligent Real-Time Simulation of Multibody Dynamics

5. FUNDING NUMBERS
DAAH04-95-1-0662

6. AUTHOR(S)
Prof. E. Bayo

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(ES)
University of La Coruña
A Maestranza s/n
15001 A CORUÑA. SPAIN

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
ARO 35002.6-EG

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document contains the Final Technical of the research project entitled Method for Intelligent Real-Time Simulation of Multibody Dynamics, which has been conducted under ARO grant number DAAH04-95- 0662.

The report includes a statament of the problems studied , the scientific accomplisments, most important results, list of publications and a list of participating scientific personnel showing the degrees earned.

The most important publications published as outcome of this research are attached to this report as an Appendix.

14. SUBJECT TERMS
influence of modelling, equation of motion and integration process in the dynamic simulation

15. NUMBER IF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

19981230 026

1. PROBLEMS STUDIED

This research was motivated by the fact that real-time simulation of large and complex multibody systems –such as Army ground vehicles– was not possible with existing technology in standard computer architectures. This was a challenge that needed to be addressed before further advances in mechanical simulation with hardware-in-the-loop and man-in-the-loop (crew station), intelligent vehicle control systems (IVCS) and virtual prototyping were made possible.

This proposed research project was aimed at developing methods and concepts that would lead to *faster-than-real-time* simulation using standard computer platforms (high-end workstations). One of the concepts proposed is that of *intelligent simulation*, which integrates, depending upon the characteristics of the system, all the factors involved in the simulation process. It was envisioned that this concept would lead to faster and more robust real-time simulators that will strongly influence the missions of the US Army TARDEC such as: intelligent vehicle control systems (IVCS), virtual prototyping and real-time simulation stations with soldier-in-the-loop. This research is in essence of multidisciplinary nature encompassing engineering mechanics, computational science and numerical methods.

More specifically the following problems have been addressed:

- Develop robust and efficient numerical algorithms in descriptor form, and hybrid integration methods for the real-time simulation of constrained mechanical systems. These algorithms will couple the dynamic formulation, kinematic constraint conditions, with hybrid integration procedures to achieve methods that will yield improved accuracy, efficiency and robustness.
- Establish the criteria and mechanisms that will allow, based on the characteristics of the system to be simulated, for the systematic choice and combination of the modeling, formulation of the equations of motion, index reduction methods, numerical integration and solution method.
- Study parallelization schemes that will integrate the different components that form the intelligent simulator in order to obtain efficient solutions faster than real-time.

2. SCIENTIFIC ACCOMPLISHMENTS AND RESULTS

2.1 Formulations and benchmark problems.

A library of benchmark problems has been created that includes the following systems: open and close chain, medium and large scale stiff problems, systems with redundant constraints, high frequency response, and singular topologies (Publications 1 and 3 below). This library has served as a testing ground for every method previously formulated or developed in the course of this research effort.

Four state of the art formulations have been implemented and tested for simulation: two augmented Lagrangian formulations in descriptor form in index-1 and index-3, a new state-space formulation suitable for stiff systems, and a new fully-recursive (order N) formulation (Publication 3).

Considering a multibody system whose configuration is characterized by n coordinates $\mathbf{q}$ that are interrelated through the m holonomic kinematic constraint conditions $\Phi(\mathbf{q}, t) = \mathbf{0}$, the equations of motion become:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{T}\,\boldsymbol{\lambda} = \mathbf{Q}(\mathbf{q},\dot{\mathbf{q}})$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{Q}$ is the vector that contains the external and non conservative forces as well as the velocity dependent inertia forces, $\Phi_{\mathbf{q}}$ is the Jacobian of the constraint equations, and $\boldsymbol{\lambda}$ is the vector that contains the Lagrange's multipliers. The last two equations constitute a set of $n+m$ mixed differential algebraic equations (DAE) of index-3, with $\mathbf{q}$ and $\boldsymbol{\lambda}$ as unknowns. This method is referred to as the classical Lagrangian formulation.

Index-1 Augmented Lagrangian Formulation with Projections

This formulation leads to the following equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{T}\boldsymbol{\alpha}\ddot{\Phi} + \Phi_{\mathbf{q}}^{T}\boldsymbol{\lambda}^{*} = \mathbf{Q}(\mathbf{q},\dot{\mathbf{q}})$$

where $\boldsymbol{\lambda}^{*}$ are the Lagrange multipliers. Introducing the expression $\ddot{\Phi} = \Phi_{\mathbf{q}}\,\ddot{\mathbf{q}} + \dot{\Phi}_{\mathbf{q}}\,\dot{\mathbf{q}} + \dot{\Phi}_{t}$ in equation the following equation is obtained

$$(\mathbf{M} + \Phi_{\mathbf{q}}^{T}\boldsymbol{\alpha}\Phi_{\mathbf{q}})\,\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{T}\boldsymbol{\lambda}^{*} = \mathbf{Q}(\mathbf{q},\dot{\mathbf{q}}) - \Phi_{\mathbf{q}}^{T}\,\boldsymbol{\alpha}\,(\dot{\Phi}_{\mathbf{q}}\,\dot{\mathbf{q}} + \dot{\Phi}_{t})$$

It is important to note that there is a substantial difference between this method and the classical Lagrange's multiplier approach presented above. The leading matrix of the classical Lagrangian method becomes singular in singular configurations, changing topologies and in the presence of redundant constraints. However, although the mass matrix $\mathbf{M}$ is in general positive semi-definite, the leading matrix of equation $(\mathbf{M} + \Phi_{\mathbf{q}}^{T}\boldsymbol{\alpha}\Phi_{\mathbf{q}})$, is always positive definite, which means that it can always be factored, even in singular positions, changing topologies and/or

with redundant constraints. The details of how the projections are performed are given in the paper attached in the Appendix.

Index-3 Augmented Lagrangian Formulation with Projections

Publication 3 (see Appendix) also presents the index-3 augmented Lagrangian method with mass-orthogonal projections. Such formulation leads to the following equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\boldsymbol{\alpha}\,\boldsymbol{\Phi} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\boldsymbol{\lambda}^{*} = \mathbf{Q}(\mathbf{q},\dot{\mathbf{q}})$$

where $\boldsymbol{\lambda}^{*}$ are the Lagrange multipliers. The numerical implementation of this formulation also leads to a similar leading matrix to that of the previous method.

Modified State-Space Formulation

A state-space representation of the equations of motion can be obtained using the basis of the null-space of the Jacobian of the constraints. Such basis is defined by the matrix $\mathbf{R}$ which will satisfy the following condition $\boldsymbol{\Phi}_{\mathbf{q}}\,\mathbf{R} = 0$. Accordingly, the dependent velocities and accelerations can be expressed in terms of the independent ones as follows:

$$\dot{\mathbf{q}} = \mathbf{R}\,\dot{\mathbf{z}} \qquad \text{and} \qquad \ddot{\mathbf{q}} = \mathbf{R}\,\ddot{\mathbf{z}} + \dot{\mathbf{R}}\,\dot{\mathbf{z}}$$

The end result (see Appendix) is the following equation:

$$\mathbf{R}^{\mathrm{T}}\mathbf{M}\,\mathbf{R}\,\ddot{\mathbf{z}} = \mathbf{R}^{\mathrm{T}}\mathbf{Q}(\mathbf{q},\dot{\mathbf{q}}) - \mathbf{R}^{\mathrm{T}}\mathbf{M}\,\dot{\mathbf{R}}\,\dot{\mathbf{z}}$$

which constitutes the equations of motion in independent coordinates. This equation provides the function evaluation for the numerical integration scheme. This method requires the solution of the position and velocity problems at each time step to obtain the matrix $\mathbf{R}$, $\mathbf{q}$ and $\dot{\mathbf{q}}$, so that the accelerations may be obtained. The solution of the position and velocity problems, that are costly computationally, have the same meaning as that of projecting the solution to obtain the dependent positions and velocities.

Implicit integrators make successive calls to the above equation as a fixed point iteration. Such iteration is either generally inefficient or precludes convergence for stiff systems. Due to this reason a method has been developed that allows this method to be used in stiff problems, and is described in detail in the publication 3 (see Appendix).

Fully-Recursive Formulation

A topological, fully-recursive algorithm of order $O(N)$ has been developed. This method is based and improves on the articulated inertia method by which accelerations are obtained in a recursive manner. The details are presented in publication 6, reproduced in the Appendix.

Comparative Results

Comparative results, illustrated in Tables I and II, lead to the following conclusions (this is the first time a comparison has been made among the most suitable methods for multibody dynamics simulation):

- The index-3 formulationis very efficient but it leads to serious ill-conditioning at small time steps, starting at about 10^{-5}, when the penalty parameter becomes problem dependent.
- The index-1 formulation with projections is the most robust of all the methods, but it may not converge for large time steps where the index-3 does. Contrary to index-3 it becomes more accurate as the time step decreases and it is not affected by ill-conditioning. It is also insensitive to the penalty parameter which remained constant in all the simulations.
- The new state-space method works well with stiffness, shows robust behavior and good energy preservation, as well as good convergence at all time steps. However, it runs considerably slower than the index-1 and index-3 methods in all the cases tested.
- The fully-recursive (articulated inertia) formulation behaves in a similar way to that of the previous one, except in what refers to speed and stiffness. This method is much faster than the state-space. Also, its improves as the size of the problem increases. For large non-stiff problems the method becomes the most competitive.
- The following tables summarize the findings of this paper. Each method is given a grade depending on its performance under different situations. The letter "A" means outstanding, "B" good, "C" fair, "D" poor and "F" signifies a failure.

			SPEED		ACCURACY		
METHOD	Easiness of Implemen-tation	Free from Graph methods	Small to Medium Problems	Medium to Large Problems	$\Delta t < 10^{-5}$	$\Delta t > 10^{-5}$ $\Delta t < 10^{-2}$	$\Delta t > 10^{-2}$
Index-1	A	A	B	B	A	A	F
Index-3	A	A	A	A	F	A	A
Space-State	B	A	C	C	A	A	A
Recursive	D	D	B	A	A	A	A

Table I. Results in Speed and Accuracy.

METHOD	PERFORMANCE				
	Changing Topologies	Singular Configurations	Inequality Constraints	Numerical Stiffness	Redundant Constraints
Index-1	A	A	A	A	A
Index-3	A	A	A	A	A
Space-State	C	F	C	A	C
Recursive	D	F	D	B	D

Table II. Performance for systems under different conditions.

- Summarizing:
  - A multi-index-1-index-3 formulation with projections is the best candidate for an efficient and robust general purpose real-time simulator.
  - The fully-recursive formulation may prove suitable for very large problems as a special purpose real-time tool.

2.2 *Modeling and performance of the Fully Recursive, Augmented Lagrangian and Classical Lagrangian methods.*

The concept of *intelligent simulation* involves the coupling of the formulations with the modeling and solution aspects. A key conclusion of this research is the importance that the modeling aspects has in the overall efficiency of the simulation process. Consequently, a description is made on some of the modeling aspects, as well as a report of the final results obtained with each of the methods including numerical stiffness in sequential as well as in a parallel processing environments.

The example used in all the recent work is the military 4x4 Bombardier Iltis vehicle whose model has been reported before. As previously described the car features four identical suspensions (see Figure 1) composed of a *shock absorber*, which provides a damping force F_D and an elastic force F_S (see Appendix). A *leaf spring*, modeled as a linear spring of 35,900 N/m. Moreover, a rubber bumper contacts the wheel at the leaf spring connection point after a vertical motion, from the equilibrium position of 70 mm. The bumper stiffness is assumed to be a constant value of 10^7 N/m, which is engaged only after the clearance is used up. This last element is considered in some simulations, thus adding a much higher stiffness to the problem. Its presence will be indicated. A *tire* is also included whose radial stiffness is 460,000 N/m. The side force and aligning torque are considered by application of the tire Calspan model. Finally, there is a steering system, which is kinematically guided during the maneuvers. The total mass of the vehicle is about 1500 Kg.

MODELING FOR LAGRANGIAN FORMULATIONS

For analysis with Lagrangian formulations, the Iltis vehicle has been modeled using fully Cartesian dependent coordinates, also called *natural* coordinates (see Appendix). For three-dimensional multibody systems, these coordinates describe the position of the whole multibody system by means of the Cartesian coordinates of *basic points* and the Cartesian components of *unit vectors*, all distributed throughout the elements. Each body of the system should have a sufficient number of points and vectors linked to it, so that its motion is completely defined. Figure 1 illustrates how the chassis, suspension and steering system of the vehicle have been modeled.

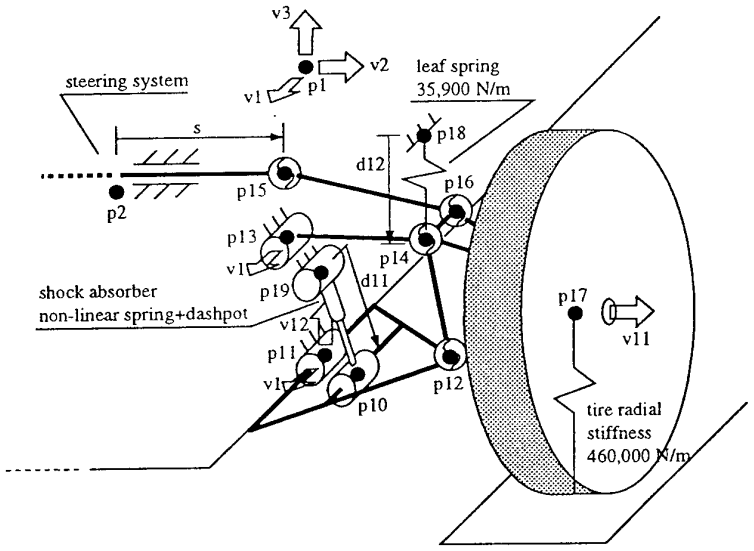


Figure 1. Modeling of the Iltis vehicle in fully Cartesian coordinates.

As may be seen in Figure 1, basic points, unit vectors and relative coordinates of distance, have been used, the total number of variables being 168. The system has 11 degrees of freedom: six free-body motion degrees of freedom for the chassis, one for each suspension and one more for the steering system. In consequence, the variables of the problem are related through 168-11=157 constraint equations. In practice, only 10 from the 11 degrees of freedom are true dynamic degrees of freedom because, as said before, the steering motion is kinematically guided. This guidance is introduced through a new constraint equation, so that the final number of constraints is 158.

The modeling problem using this type of coordinates becomes straight-forward and it may be used in general purpose simulators, in fact its use is very similar to that of the finite element method in structural mechanics.

MODELING FOR RECURSIVE FORMULATION

Recursive formulations require the use of relative coordinates. They define the position of each body in relation to the previous one in the kinematic chain, by using the parameters or coordinates corresponding to the relative degrees of freedom allowed by the joint linking those elements. Figure 2 illustrates the way the chassis, suspension and steering system of the Iltis vehicle have been modeled using this kind of coordinates.

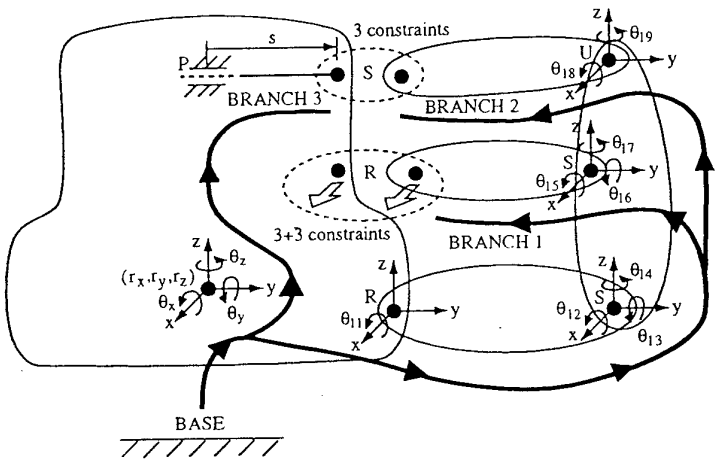


Figure 2. Modeling of the Iltis vehicle in relative coordinates.

In this case, the number of variables is 43, distributed as follows: 6 for the chassis (three translations and three rotations), 9 for each suspension (all of them rotations at the different pairs as shown in Figure 2) and 1 distance for the steering system. As this last variable is kinematically guided, the total number of variables that are integrated is 42. Figure 2 shows where the closed-loops have been cut. Two cuts are performed for each suspension. One of them needs 3+3=6 constraints to establish identity between points and directions at the rotational joint removed (actually, only two constraints are needed to specify the direction but three makes it easier), while the other only introduces 3 more constraints as this time the joint removed is spherical. Therefore, the total number of constraints is 36, with just 32 independent.

The modeling problem using this type of coordinates is quite involved and requires a substantial effort particularly in the existence of close loops. Its use is not recommended for general purpose simulators, but for special ones that would allow for a time consuming effort of implementation.

DESCRIPTION OF THE SIMULATIONS

The first simulation, called SIMUL1, consists of 10 seconds of motion with the vehicle going up an inclined ramp and the down a series of stairs at a speed of 5 m/s, as may be seen in Figure 3. The simulation leads to the strong motion with accelerations of up to 5*g*.

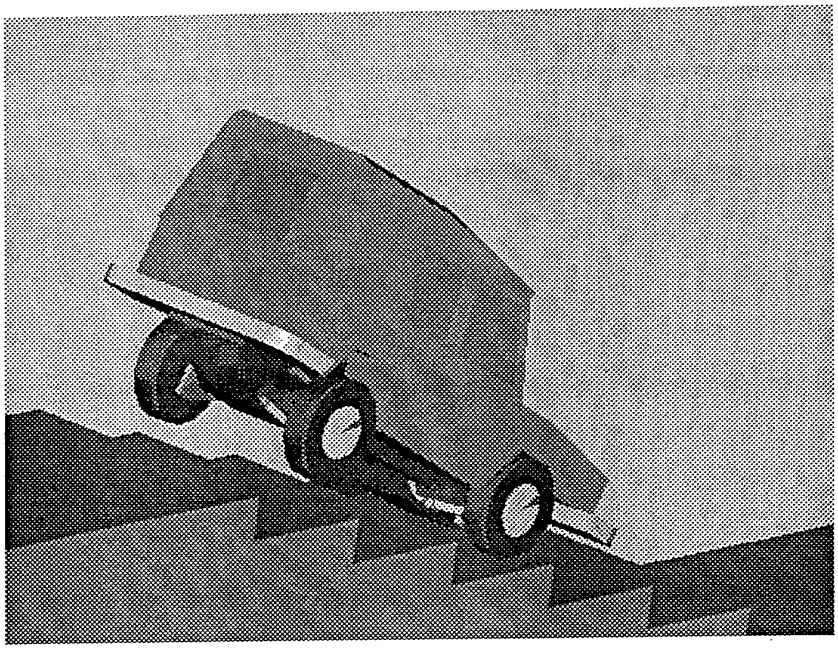


Figure 3. The Iltis vehicle performing SIMUL1.

During the second simulation, SIMUL2, that also lasts for 10 seconds at a speed of 5 m/s, the vehicle goes off a ramp and hits the ground as illustrated in Figure 4.

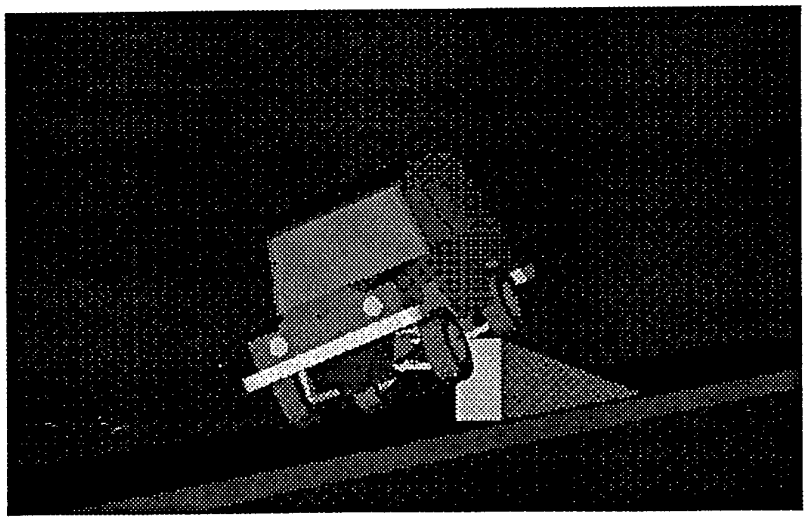


Figure 4. The Iltis vehicle performing SIMUL2.

In the third simulation, SIMUL3, the Iltis vehicle is driven through a series of columns in the slalom maneuver shown in Figure 5. The simulation lasts for 10 seconds at a vehicle speed of 5 m/s.

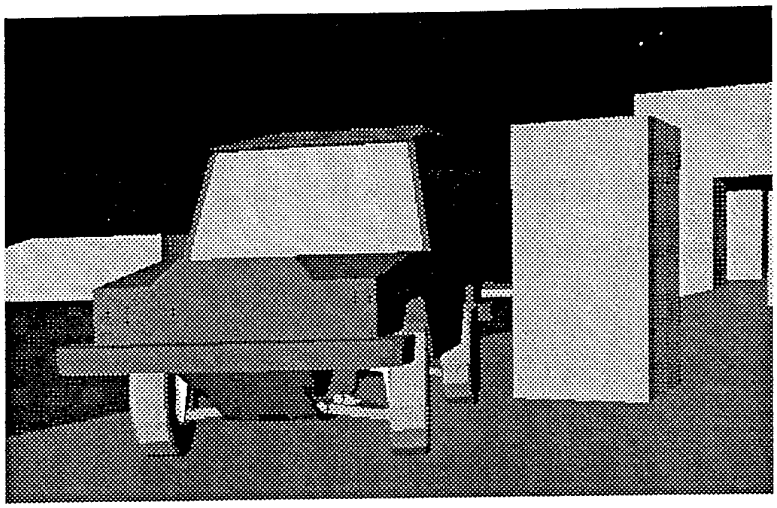


Figure 5. The Iltis vehicle performing SIMUL3.

RESULTS ON ONE-PROCESSOR MACHINES. SEQUENTIAL PROCESSING.

The three simulations described above were carried out on a SGI Indigo2 IMPACT with one processor R4400SC @ 200 MHz and 2 Mb of secondary cache memory, using the three formulations: Augmented Lagrangian, Classical Lagrangian and Fully Recursive. Table I features two columns for each formulation: the first one gives the smallest CPU time achieved with the corresponding method, while the second one shows the time step size which led to the best CPU time.

Table III illustrates the fact that the augmented Lagrangian formulation (with a penalty factor of 10^9) is clearly superior to the classical one because it performs notably faster (between three and four times) and achieves convergence for greater time step sizes. On the other hand, the fully-recursive formulation shows to be more efficient than the augmented Lagrangian method although it needs to take a smaller time step size to converge.

Therefore, when the simulation difficulty is not very high and the augmented Lagrangian method can work with large time step sizes (SIMUL2), it performs at the same level that the fully-recursive formulation. On the contrary, when the maneuver becomes sharper (SIMUL1) the augmented Lagrangian method must keep on small step sizes and the fully-recursive formulation takes a fair advantage.

	Augmented Lagrangian		Classical Lagrangian		Fully-Recursive	
	CPU (s)	Δt (s)	CPU (s)	Δt (s)	CPU (s)	Δt (s)
SIMUL1	**34**	0.0175	**139**	0.01	**20**	0.0075
SIMUL2	**18**	0.03	**58**	0.025	**14**	0.0075
SIMUL3	**27**	0.025	**99**	0.01	**9**	0.0075

Table III. Efficiency on a single-processor machine.

RESULTS ON PARALLEL MACHINES

To find out the effect of the parallelization over the three formulations under study, simulation SIMUL1 has been executed on a SUN Sparc Station 20 with 4 processors HyperSparc (RT625) @ 100 MHz with 256 Kb of cache memory. Table IV shows the reduction in CPU time experienced by each method as the number of processors increases.

# of processors	Augmented Lagrangian	Classical Lagrangian	Fully-Recursive
1	0.0	0.0	0.0
2	22.7	6.5	0.0
3	29.6	9.0	0.0
4	29.6	9.0	0.0

Table IV. Reduction of CPU time in % due to parallelization.

It may be seen from Table IV that the method that takes most advantage of the parallelization, up to 30% savings, is the augmented Lagrangian, since it uses it during the matrix formation and solution. The classical Lagrangian only takes advantage of the parallelization at the time of solving the equations, achieving up to 9% reduction. On the other hand, the fully-recursive formulation does not profit at all from the parallel computing, thus indicating to be a method suitable for sequential processing. It must be noticed that, in all cases, the fourth processor does not contribute to make any difference in CPU time.

The simulations SIMUL1 and SIMUL2 are also performed in an ALPHA AXP 4000 with 2 processors Alpha 5 @ 466 MHz with 4Mb of cache memory. The performance of the augmented Lagrangian method is shown in Table V. Real-time performance is achieved for SIMUL2, thus corroborating the fact that real-time simulation can be achieved in current standard workstations using the augmented Lagrangian formulation.

Number of Processors	SIMUL1	SIMUL2
2	12 seconds	9 seconds

Table V. CPU times taken by the augmented Lagrangian method in the ALPHA AXP.

INFLUENCE OF NUMERICAL STIFFNESS

Simulation SIMUL1 is carried out again using the three formulations. However, this time the rubber bumpers of stiffness 10^7 N/m mentioned above, are activated in the four suspensions of the model, so that the behavior of the algorithms under conditions of high numerical stiffness may be observed. This simulation is now called SIMUL1S. Table VI compares the results obtained by the three methods when performing SIMUL1 and SIMUL1S.

	Augmented Lagrangian		Classical Lagrangian		Fully-Recursive	
	CPU (s)	Δt (s)	CPU (s)	Δt (s)	CPU (s)	Δt (s)
SIMUL1	**34**	0.0175	**139**	0.01	**20**	0.0075
SIMUL1S	**78**	0.005	**435**	0.0025	**34**	0.0025

Table VI. Influence of numerical stiffness.

Table VI shows that numerical stiffness makes the Lagrangian methods take a small time step size, thus loosing performance with respect to the fully-recursive formulation. On the other hand, the efficiency ratio between the augmented and classical Lagrangian methods does not seem to change. However, the penalty factor of the augmented Lagrangian procedure needs to be increased to 10^{10} to achieve convergence and, at lower time steps, as for instance 10^{-3} seconds, this formulation starts suffering from its characteristic ill-conditioning effect. When numerical stiffness remains low, this problem arises at a step size of 10^{-4} or 10^{-5} seconds, but the presence of high numerical stiffness makes it to appear sooner.

CONCLUSIONS

This is the first time that a thorough comparison of different leading methods of dynamic simulation has been carried out for large multibody systems. The conclusions that have been reached may be summarized as follows:

- The modeling process required by the fully-recursive method leads to a relative coordinate formulation that is suitable for open chain configurations, but presents serious complexities and difficulties for the case of closed kinematic chains. The modeling for the Lagrangian formulations is much simpler and may be done with dependent coordinates. Therefore the former calls for special purpose implementations, whereas the latter is more suitable for

2.3 *Refined strategy for a multi-index variable time step integration method.*

A refined strategy has been implemented for the multi-index variable time step that has led to a better behavior and control of the time step size during the integration process (publications 2 and 5). A summary of the method and results are reported below (see Appendix for more details).

THE MULTI INDEX AUTOSTEPPING ALGORITHM

As reported above the augmented Lagrangian index-1 method is more accurate than the index-3 for small time steps, and the index-3 method is more efficient than the index-1 (with same accuracy) for large time steps. Therefore, a multi-index formulation would take advantage of the good qualities of both methods for different time step sizes. Index-3 is the right choice for time steps between 10^{-2} and 10^{-4} (where it is accurate and efficient); however, as we decrease the size of the time step, starting in the neighborhood of $\Delta t = 10^{-4}$ (with no pivoting) or $\Delta t = 10^{-5}$ (with pivoting). the index-1 method becomes more accurate and robust.

Also, taking into account the correlation existing between the energy and the local error an integral of motion (system total energy in conservative systems) has been proposed as a measure of the local integration error. For non-conservative systems and considering the use of fully Cartesian coordinates, the following integral of motion (energy invariant) is established:

$$\Pi = T(t) + V(t) - \int_0^t \dot{\mathbf{q}}^T \mathbf{Q} d\tau = constant$$

where T is the kinetic energy and V the potential energy, respectively. This equation clearly yields $T + V = constant$ for conservative systems.

Using this energy invariant described as a measure of the local integration error, a variable time step size strategy has been proposed. The key idea is to modify the step size when the change in the energy invariant exceeds an allowable value. When this occurs a new time step size is calculated in order to achieve the desired accuracy.

NUMERICAL RESULTS

Simulations are performed using the front suspension of the ILTIS vehicle, one of the models included in the library of benchmark problems. The model requires a total of 23 variables, related through 22 constraint equations.

The simulation is performed as follows: the suspension starts moving from at rest conditions at a speed of 5 m/s, and its position does not correspond to the static equilibrium. Thus, it freely oscillates until the static equilibrium position is reached. After three seconds, the suspension goes over a road bump, defined by a cosine function, and afterwards it freely oscillates until the equilibrium is reached again. The complete analysis lasts for 6 seconds.

general purpose simulators. In this respect, an analogy may be established between these methods and the finite element formulation for structural analysis.

- The augmented Lagrangian method (ALF) features a leading matrix that is symmetric and positive-definite. This makes it very robust in simulations with singular configurations and redundant constraints. Also, this method permits the differentiation of the dynamic equations to get the tangent matrix in a Newton-Raphson iteration, which allows for convergence at larger time step sizes than the recursive formulations. Furthermore, it takes a serious advantage of parallel computing. However, it suffers from ill-conditioning when the time step size becomes small, thus having convergence difficulties for stiff systems. A precision environment of 32-digits (words of 128 bits) could be the cure for this numerical drawback.

- The classical Lagrangian formulation generates an almost double-size problem than its augmented counterpart, and takes less profit of parallel computing, thus, performing notably worse. In addition, its leading matrix is less robust, causing the failure of the integration process at singular configurations and in the presence of redundant constraints. However, this method is not so sensitive to ill-conditioning problems for small time step sizes.

- The fully-recursive procedure performs well for stiff and non-stiff systems. Its involved formulation only permits a fixed-point iteration scheme, so that the time step size must be kept small to achieve convergence. Despite of this fact, it shows to be the most efficient method among the three compared, but does not obtain any advantage from parallel computing. The integration also fails in singular configurations and in the presence of redundant constraints.

- According to the presented results, real-time simulation of large multibody systems can be achieved with medium-size workstations, opening a wide field for development of hardware-in-the-loop and man-in-the-loop facilities, virtual prototyping tools and intelligent vehicle control systems.

This simulation is carried out under two different conditions. The first one is done using a index-3 approach with a fixed time step of 10^{-2} seconds. The second simulation is performed at variable time step and multi-index, starting with index-3 and a step size of 10^{-3} seconds. The first simulation took 8.41 seconds of CPU time and the second took 36.07 seconds.

Figure 6 shows the time history of the time step size. It may be clearly seen that, for the specified tolerance it is necessary to shorten the time step size only when the wheel is passing over the bump. At the beginning the step size is quickly adjusted to speed up the process while maintaining the local error between the allowable limits.

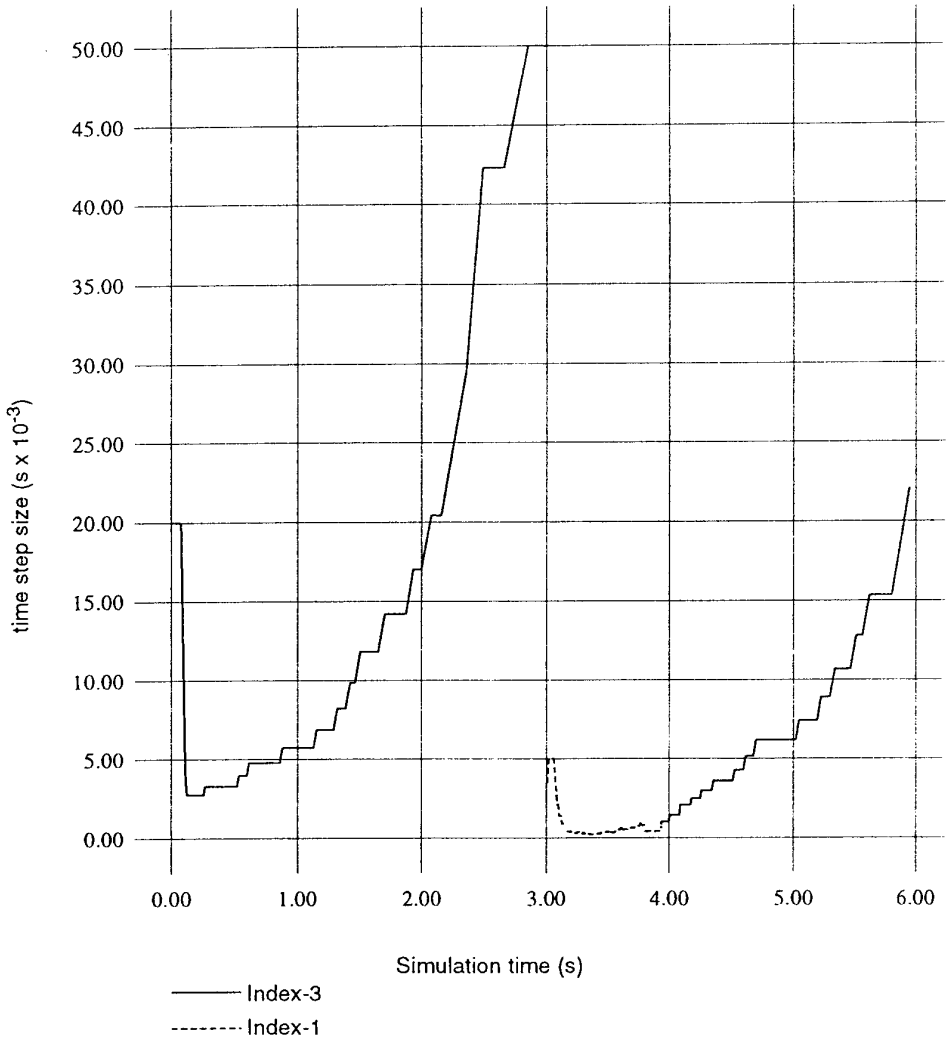


Figure 6. Time history of step size.

Figure 7 shows the total system energy in both cases. It is important to remark that only when translating the origin and scaling the energy represented in the vertical axis one can notice its change. The biggest deviation from the initial value (425,128 J vs. 100 J) is about 0.02% of the total system energy in both cases. Also important it is the fact that similar precision in total

system energy conservation can be obtained performing the simulation with an index-1 approach and a fixed time step size of 5.10^{-4} seconds. In that case the simulation takes 155 seconds (over 400% more than in the variable index-variable step case), thus corroborating the benefits of using the proposed multi-index variable time step procedure.

As a result of this example, it is clear that a multi-index-1-3 approach with variable time step provides an excellent solution, since the more efficient index-3 method is used during a large part of the motion, and the index-1 only at that part where the time step decreases below the critical limit where the index-3 loses precision and accuracy.

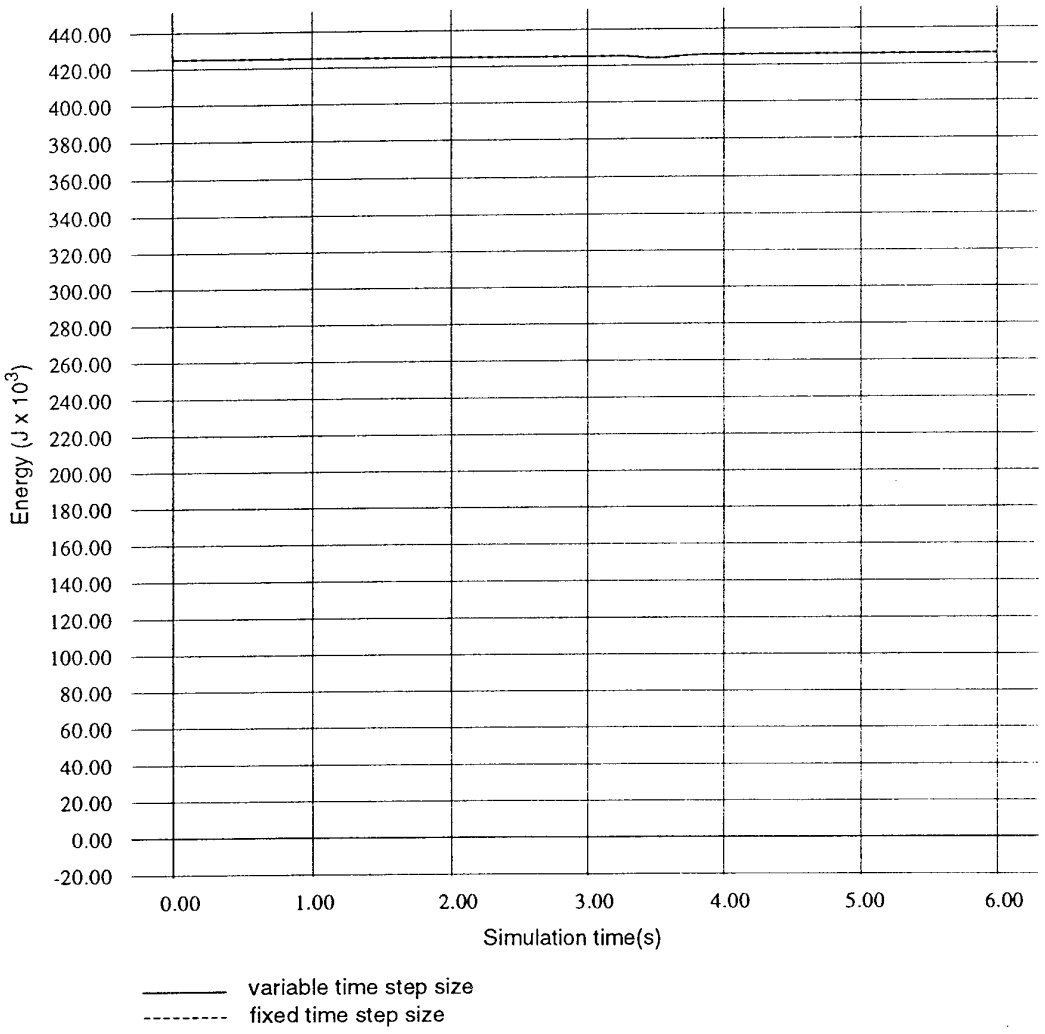


Figure 7. Total system energy.

CONCLUSIONS

- The system energy invariant has proved to be a good measure of the local integration error. This feature makes unnecessary the use of traditional (but less efficient) error criteria used in standard numerical integration methods.
- The variable time step strategy and the switch from one method to the other do not cause problems during the integration process.
- The use of the proposed technique allows a substantial speedup gain in CPU time for large scale systems, thus helping to achieve real-time behavior.
- The method is general and can also be applied to solve the dynamics of flexible multibodies.

3. *PUBLICATIONS*

ARCHIVE PUBLICATIONS

1. Cuadrado, J., Cardenal, J. and Bayo, E., "Modeling issues for the intelligent simulation of multibody systems". Proceedings of 24th ASME Biennial Mechanisms Conference, MECH-1011. Irvine, CA. Aug. 1996.

2. Cardenal, J., Cuadrado, J. and Bayo, E., "A multi-index variable time step method for the dynamic simulation of multibody systems". Proceedings of 16th ASME Computers in Engineering Conference, CIE-1625. Irvine, CA. Aug. 1996.

3. Cuadrado, J., Cardenal, J. and Bayo, E. "Modeling and solution methods for efficient real-time simulation of multibody dynamics", *Journal of Multibody System Dynamics.* Vol. 1, No. 3, pp. 259-280. Sept. 1997.

4. Cardenal, J., Cuadrado, J. and Bayo, E. "A strategy to accelerate the numerical integration in the dynamic simulation of multibody systems", Paper VIB-4228. ASME 16th Biennial Conference on Mechanical Vibration and Noise. Symposium on Multibody Dynamics and Vibration. Sacramento, California. Sept. 1997.

5. Cardenal, J., Cuadrado. J, Morer, P. and Bayo, E. "A Multi-Index Variable Time Step Method for the Dynamic Simulation of Multibody Systems". Accepted for publication *Inter. Journal Numerical Methods in Engineering.* Vol. 44. 1999.

6. Cuadrado, J., Cardenal, J., Morer, P. and Bayo, E. "Intelligent simulation of multibody dynamics: Space-state and descriptor methods in sequential and parallel computing". Submitted to the *Journal of Multibody System Dynamics.*

REPORTS

1. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". First Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. Jan. 1996.

2. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Second Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. April 1996.

3. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Third Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. July 1996.

4. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Fourth Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. Oct. 1998.

5. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Fifth Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. Jan. 1997.

6. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Sixth Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. April 1997.

7. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Seventh Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. July 1997.

8. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Eight Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. Oct. 1997.

9. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Nineth Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. Jan. 1998.

10. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Tenth Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. April 1998.

11. Bayo, E. "Methods for Intelligent Real-Time Simulation of Multibody Dynamics". Eleventh Technical Progress Report. Mechanical Engineering Department. Universidad de la Coruña. July 1998.

4. SCIENTIFIC PERSONNEL

Prof. Eduardo Bayo. Principal Investigator.

Dr. Javier Cuadrado. Post.Doctoral Researcher advanced to Assistant Professor.

Dr. Jesús Cardenal. Post Doctoral Researcher advanced to Assistant Professor.

Ms. Paz Morer. Graduate student advanced to Ph.D.

Ms. Ruth Gutiérrez. Graduate student advanced to Ph.D. Candidate.

Mr. Pablo de Castro. Undergraduate student advanced to Ph.D. program.

Mr. Julio Díaz. Undergraduate student advanced to Ph.D. program.

Mr. Alfonso Loureiro. Undergraduate student advanced to Ph.D. program.

APPENDIX

PUBLICATION 1

Multibody System Dynamics **1**: 259–280, 1997.

Modeling and Solution Methods for Efficient Real-Time Simulation of Multibody Dynamics

J. CUADRADO and J. CARDENAL
Escuela Politécnica Superior, University of La Coruña, Mendizábal, s/n, 15403 Ferrol, Spain

E. BAYO
Escuela Politécnica Superior, University of La Coruña, Mendizábal, s/n, 15403 Ferrol, Spain; and School of Engineering and Architecture, University of Navarre, 31080 Pamplona, Spain

(Received: 28 October 1996; accepted in revised form: 4 April 1997)

Abstract. Current simulation tools for multibody dynamics are not problem dependent, they use the same modeling process to all cases regardless of their characteristics. In addition, real-time simulation of small multibody systems is achievable by existing simulation tools, however, real-time simulation of large and complex systems is not possible with existing methods. This is a challenge that needs to be addressed before further advances in mechanical simulation with hardware-in-the-loop and man-in-the-loop, as well as virtual prototyping are made possible.

This paper addresses the issue of how the modeling process – dependent *versus* independent co-ordinates, and descriptor form *versus* state-space form of the equations of motion – affects the dynamic simulation of multibody systems and how it could be taken into account to define the concept of intelligent simulation. With this new concept all the factors involved in the simulation process – modeling, equations, solution, etc. – are chosen and combined depending upon the characteristics of the system to be simulated. It is envisioned that this concept will lead to faster and more robust real-time simulators.

Key words: real-time-simulation, computational multibody dynamics, solution algorithms.

1. Introduction and Background

The kinematics and dynamics of multibody systems constitute an important part of what is referred to as CAD and MCAE. The advantage of computer simulations performed by CAD and MCAE tools, is that they allow designers to predict the kinematic and dynamic behavior of all types of multibody systems in great detail, during all the design stages: from the first design concepts to the final prototypes. The kinematic and dynamic analysis of multibody systems are processes which are most appropriately performed in *real-time*, allowing even the analyst to act as an additional element in the simulation process, *man-in-the-loop*, by introducing external or control forces over specific degrees of freedom. This obviously imposes constraints on the computer hardware and software. Real-time analysis now requires the use of, at least, the top of the line workstations, and is not yet possible for large problems.

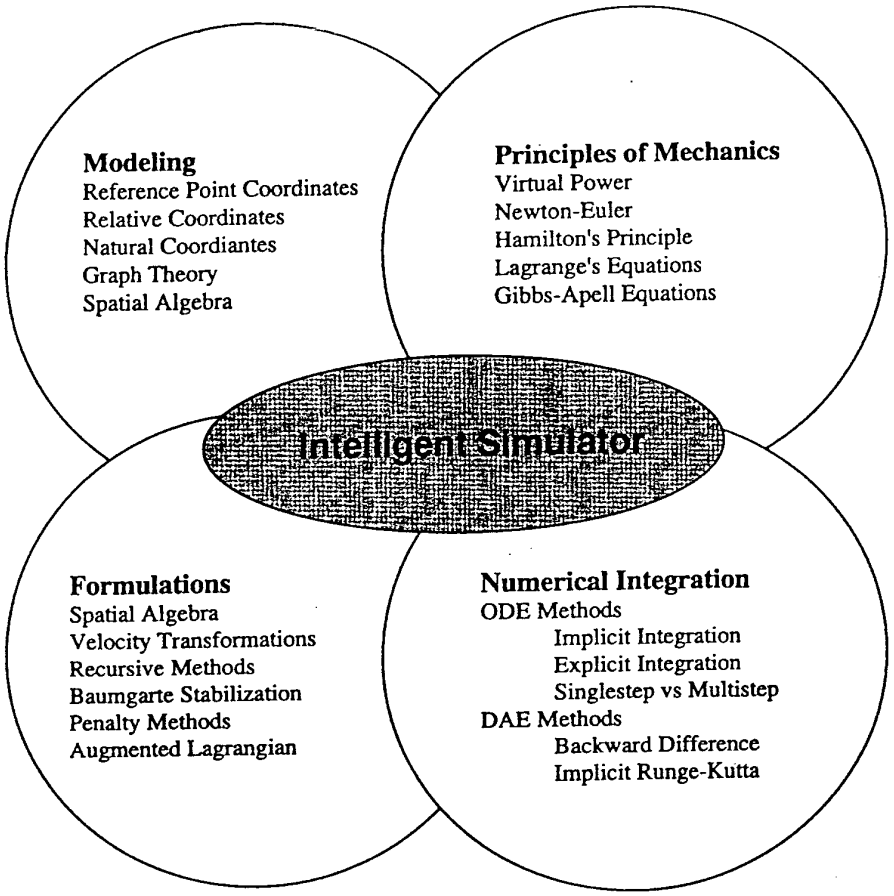


Figure 1. Factors involved in the simulation process.

Figure 1 illustrates all the factors that intervene in a simulation process. Such factors include: modeling process, use of the principle of mechanics, formulation of the equations of motion, numerical integration and solution of the resulting equations. Intelligent simulation requires the right choice of these factors which will lead to the best possible simulation of the multibody system at hand.

The first issue to be considered in the simulation is that of modeling the system, that is, the selection of a set of parameters or co-ordinates that will allow to unequivocally define at all times the position, velocity and acceleration of the system. The most important types of co-ordinates currently used to define the motion of multibody systems are: *relative* co-ordinates, *reference point* or *Cartesian* co-ordinates, and *natural* or *fully Cartesian* co-ordinates. A thorough comparison between these different types of co-ordinates and their influence in the solution process has been carried out by García de Jalón and Bayo [1]. As a result of that study, we choose in this paper the fully Cartesian co-ordinates which lead to constant inertia terms and linear Jacobian of the kinematic constraints, and are numerically more efficient than the reference point co-ordinates [1].

Another factor in the modeling process is the choice of the dynamic formulation, obtained from the application of the principles in dynamics, that will lead to the final form of the equations of motion. Dynamic principles such as Lagrange's equations, Newton's laws, canonical equations of Hamilton, virtual power, Hamilton's principle, Gibbs–Apell equations, etc. constitute the basis for the formulations of multibody dynamics. The choice of the dynamic formulation will influence the subsequent choice of numerical integration schemes.

The widely used method of Lagrange's multipliers leads to a representation of the equations of motion in its *descriptor form* constituting a set of index-3 differential algebraic equations (DAE). The addition of stabilization techniques, such as the method of Baumgarte [2], reduces the index and makes the solution tractable by means of standard ODE solvers, however, this method does not provide full constraint satisfaction, leads to a limited control of accuracy, and in addition no ways are available for choosing the values of the coefficients used by the method. Recently, an augmented Lagrangian formulation with projections [3] was proposed that in addition to transforming the set of equations into a stabilized set, is solvable by standard ODE methods and assure Lyapunov stability of the simulation process [4]. They also have the advantages of being robust under singular configurations, topology changes and with redundant constraints and provide full constraint satisfaction.

Other formulations, such as co-ordinate partitioning [5], Kane's method [6], and virtual power with projection matrices [7] transform the equations of motion to a minimum set of co-ordinates or *state-space form* that is directly solvable by ODE methods. State-space representations may also be obtained by means of *velocity transformations*, initially introduced by Jerkovsky [8], and subsequently extended by other authors [9–13]. Recursive methods, also called topological since they take into account the structure of the mechanism, such as those developed by Walker and Orin [14] or Featherstone [15, 16], are also included in this family. State-space representations are more suitable for ODE integration than the descriptor counterparts at the expense of solving the velocity and position problem at each time step. However, they do not handle well topology changes and singular configurations. In addition, they cannot support stiff integrators and consequently may present difficulties when the system has built-in numerical stiffness.

The numerical mathematics community has sought solutions to the index reduction problem and has proposed many different ways. Recent advances have been made which have yielded stabilized index reduction methods and accurate ways of projecting the DAE onto the underlying ODE for more stable and accurate solutions. Key developments are the work of Brenan et al. [17], Griepentrog et al. [18], Führer and Leimkuhler [19], Lubich et al. [20], developer of MEXX, Arnold [21], developer of HEDOPS, Hairer and Wanner [22], developers od RADAU5, and Petzold [23], developer of DASSL.

In this paper we examine and compare four methods: a recently proposed augmented Lagrangian formulation with projections [3] in *index-1* and *index-3*, a

new *modified state-space* formulation based on projection matrices, that is capable of handling the numerical stiffness coming from built in stiff numerical properties (springs, dashpots, etc.), and a fully-recursive formulation proposed by Jiménez [24] which is an improvement of the *articulated inertia* method [15]. The goal is to be able to find guidelines as to what modeling methods – dependent *versus* independent co-ordinates, descriptor form *versus* state-space form of the equations of motion, global *versus* topological methods – are most adequate for different types of multibody systems. For this reason, the comparison is made through the analysis of different mechanical systems that encompass: open and close chains, systems that undergo singular configurations, modeling with redundant constraints, systems that encounter sudden changes and high frequency thus requiring very small time steps of integration, and lastly numerically stiff systems. Comparisons are made on the basis of qualitative as well as quantitative factors. Conclusions are drawn based on the performance of the different methods that serve as the basis to establish a strategy for intelligent simulation.

2. Modeling and Dynamic Analysis

2.1. Formulation in Fully Cartesian Co-ordinates

Let us consider a multibody system whose configuration is characterized by n fully Cartesian (or natural) co-ordinates $\mathbf{q}$ [1] that are interrelated through the m holonomic kinematic constraint conditions

$$\Phi(\mathbf{q}, t) = 0. \tag{1}$$

The use of the principle of virtual power directly leads to the equations of motion

$$\delta\dot{\mathbf{q}}^T(\mathbf{M}\ddot{\mathbf{q}} - \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) + \Phi_q^T\boldsymbol{\lambda}) = 0, \tag{2}$$

which for a general multibody system leads to:

$$\mathbf{M}\ddot{\mathbf{q}} + \Phi_q^T\boldsymbol{\lambda} = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}), \tag{3}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{Q}$ is the vector that contains the external and non-conservative forces as well as the velocity dependent inertia forces, Φ_q is the Jacobian of the constraint equations, and $\boldsymbol{\lambda}$ is the vector that contains the Lagrange multipliers. Note that the use of the fully Cartesian co-ordinates leads to a constant mass matrix $\mathbf{M}$ and the absence of velocity dependent inertia forces in the vector $\mathbf{Q}$, and consequently Equation (3) is greatly simplified.

Equations (1) and (3) constitute a set of $n + m$ mixed differential algebraic equations (DAE) of index-3 [17], with $\mathbf{q}$ and λ as unknowns. It is a common practice in multibody dynamics to double differentiate the constraints and append the resulting equations to (3) to avoid the direct integration of DAEs, thus yielding an index-1 formulation:

$$\begin{bmatrix} \mathbf{M} & \Phi_q^T \\ \Phi_q & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ -\dot{\Phi}_q\dot{\mathbf{q}} - \dot{\Phi}_t \end{Bmatrix}. \tag{4}$$

These equations can now be integrated using standard numerical integrators with each function evaluation performed using Equation (4).

2.2. INDEX-1 AUGMENTED LAGRANGIAN FORMULATION WITH PROJECTIONS

Recently, a method was presented by Bayo and Ledesma [3] that uses an index-1 augmented Lagrangian method with mass-orthogonal projections of the positions and velocities to their constraint manifolds. Such formulation leads to the following equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \Phi_q^T \alpha \ddot{\Phi} + \Phi_q^T \lambda^* = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}), \tag{5}$$

where λ^* are the Lagrange multipliers. Introducing the expression $\ddot{\Phi} = \Phi_q \ddot{\mathbf{q}} + \dot{\Phi}_q \dot{\mathbf{q}} + \dot{\Phi}_t$ in Equation (5) the following equation is obtained

$$(\mathbf{M} + \Phi_q^T \alpha \Phi_q)\ddot{\mathbf{q}} + \Phi_q^T \lambda^* = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) - \Phi_q^T \alpha(\dot{\Phi}_q \dot{\mathbf{q}} + \dot{\Phi}_t). \tag{6}$$

It is important to note that there is a substantial difference between Equation (6) and the Lagrange's multiplier approach represented by (4). The leading matrix of Equation (4) becomes singular in singular configurations, changing topologies and in the presence of redundant constraints. However, although the mass matrix $\mathbf{M}$ is in general positive semi-definite, the leading matrix of Equation (6), $(\mathbf{M} + \Phi_q^T \alpha \Phi_q)$, is always positive definite, which means that it can always be factored, even in singular positions, changing topologies and/or with redundant constraints.

The following iteration process [3] yields the unknown multipliers λ^*

$$\lambda^*_{i+1} = \lambda^*_i + \alpha(\ddot{\Phi})_{i+1}, \quad i = 0, 1, 2, \ldots \tag{7}$$

with $\lambda^*_0 = 0$ for the first iteration. Equation (7) physically represents the introduction at iteration $i + 1$ of forces that tend to compensate the fact that the addition of all the constraint terms are not exactly zero. It turns out that with the augmented Lagrangian formulation, the penalty numbers do not need to be very large (thus leading to a better numerical conditioning) since the resulting error in the constraint equations will be eliminated by the Lagrange terms during the iteration procedure. The value of the penalty factor α does not affect the solution, but only the convergence rate. Experience shows that when the constraints are scaled to unity, penalty factors ranging from 10^5 to 10^7 give a good convergence rate, and only 1 to 2 iterations are required to converge to the solution. The leading matrix remains constant during the iteration process.

As a result of using the index-1 formulation, the solution of Equation (7) yields a set of accelerations that not only satisfies dynamic equilibrium but also the constraint conditions $\ddot{\Phi} = \mathbf{0}$. However, it is expected that some constraint violations will take place for $\Phi = \mathbf{0}$ and $\dot{\Phi} = \mathbf{0}$. A projection scheme is devised for $\mathbf{q}$ and $\dot{\mathbf{q}}$ such that the constraints Φ and $\dot{\Phi}$ will be zero also. These projections may be done as follows:

(a) Projection in $\mathbf{q}$.
During the time integration process the numerical integration scheme yields a set of co-ordinates $\mathbf{q}^*$ that does not completely satisfy the constraint conditions $\Phi = \mathbf{0}$. In order to satisfy the constraints, we perform a mass-orthogonal projection of the solution to the constraint manifold and obtain a new set of positions $\mathbf{q}$ that satisfies $\Phi = \mathbf{0}$. This can be achieved by the solution of the following iterative procedure [3]:

$$(\mathbf{M} + \Phi_q^T \alpha \Phi_q) \Delta \mathbf{q}^{(i+1)} = -\gamma_p^{(i)} \tag{8}$$

with

$$\gamma_p^{(i)} = \mathbf{M}(\mathbf{q}^{(i)} - \mathbf{q}^*) + \Phi_q^T \boldsymbol{\lambda}^{(i)} \tag{9}$$

and

$$\mathbf{q}^{(i+1)} = \mathbf{q}^{(i)} + \Delta \mathbf{q}^{(i+1)}, \quad \boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} + \alpha \Phi^{(i+1)}, \tag{10}$$

where the superscript indicates the iteration number. Equations (8) through (10) can be used iteratively until $\|\Delta \mathbf{q}\| < \varepsilon$, where ε is a user specified tolerance.

Note the important feature that the tangent matrix in Equation (8) is identical to that used in Equation (6) for the dynamic analysis. Furthermore, since the solution $\mathbf{q}$ is close to the initial values $\mathbf{q}^*$, the projection problem can be solved using a modified Newton–Raphson method with no need for updating the tangent matrix.

(b) Projection in $\dot{\mathbf{q}}$.
Similarly during the time marching integration process the numerical integration scheme yields a set of velocities $\dot{\mathbf{q}}^*$ that does not completely satisfy the constraint conditions $\dot{\Phi} = \mathbf{0}$. Again, we perform a mass-orthogonal projection of the solution to the velocity constraint manifold and obtain a new set of velocities $\dot{\mathbf{q}}$ that will satisfy $\dot{\Phi} = \mathbf{0}$. This can be achieved by the solution of the following equation [3]:

$$[\mathbf{M} + \Phi_q^T \alpha \Phi_q] \dot{\mathbf{q}} = \mathbf{M} \dot{\mathbf{q}}^* - \Phi_q^T (\alpha \Phi_t + \sigma), \tag{11}$$

where the Lagrange multipliers of the projection problem are updated as:

$$\sigma^{(i+1)} = \sigma^{(i)} + \alpha \dot{\Phi}^{(i+1)}. \tag{12}$$

A more efficient implementation of the projection in velocities is given by the following recursive set of equations:

$$[\mathbf{M} + \Phi_q^T \alpha \Phi_q] \dot{\mathbf{q}}^{(i+1)} = \mathbf{M} \dot{\mathbf{q}}^{(i)} - \Phi_q^T \alpha \Phi_t, \tag{13}$$

and the iteration is started by setting

$$\dot{\mathbf{q}}^{(0)} = \dot{\mathbf{q}}^* \quad \text{and} \quad \sigma^{(0)} = \alpha \dot{\Phi}^{(0)}. \tag{14}$$

Since the leading matrix in Equation (11) depends only on the generalized co-ordinates, it needs to be triangularized only once, and the iteration described

by Equation (13) will require only successive forward reductions and back-substitutions.

2.3. INDEX-3 AUGMENTED LAGRANGIAN FORMULATION WITH PROJECTIONS

In [3], an index-3 augmented Lagrangian method with mass-orthogonal projections of the velocities and accelerations to their constraint manifolds is presented. Such a formulation leads to the following equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha} \boldsymbol{\Phi} + \boldsymbol{\Phi}_q^T \boldsymbol{\lambda}^* = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}), \tag{15}$$

where $\boldsymbol{\lambda}^*$ are the Lagrange multipliers. The following iteration process yields the unknown multipliers $\boldsymbol{\lambda}^*$

$$\boldsymbol{\lambda}_{i+1}^* = \boldsymbol{\lambda}_i^* + \boldsymbol{\alpha} \boldsymbol{\Phi}_{i+1}, \quad i = 0, 1, 2, \ldots \tag{16}$$

with $\boldsymbol{\lambda}_0^* = 0$ for the first iteration. Similar to the case of the index-1 formulation the value of the penalty factor $\boldsymbol{\alpha}$ affects the convergence rate. Experience shows that when the constraints are scaled to unity, penalty factors ranging from 10^7 to 10^8 give a good convergence rate for the index-3 formulation, and only 1 to 2 iterations are required to converge to the solution.

As a result of using the index-3 formulation, the solution of Equation (15) yields a set of $\mathbf{q}_{n+1}$ that not only satisfies dynamic equilibrium but also the constraint conditions $\boldsymbol{\Phi} = \mathbf{0}$. However, it is expected that some constraint violations will take place for $\dot{\boldsymbol{\Phi}} = \mathbf{0}$ and $\ddot{\boldsymbol{\Phi}} = \mathbf{0}$. A projection scheme is devised for $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ such that the constraints $\dot{\boldsymbol{\Phi}}$ and $\ddot{\boldsymbol{\Phi}}$ will be zero also. The projection in $\ddot{\mathbf{q}}$ is done as follows:

(c) Projections in $\ddot{\mathbf{q}}$.
Similar to the velocity analysis, the projection of the generalized accelerations onto the constraint manifold can be obtained through the solution of the following equation [3]:

$$(\mathbf{M} + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha} \boldsymbol{\Phi}_q)\ddot{\mathbf{q}} = \mathbf{M}\ddot{\mathbf{q}}^* - \boldsymbol{\Phi}_q^T \{\boldsymbol{\alpha}(\dot{\boldsymbol{\Phi}}_q \dot{\mathbf{q}} + \boldsymbol{\Phi}_t) + \boldsymbol{\kappa}\}, \tag{17}$$

where the associated Lagrange multipliers are updated by the following formula:

$$\boldsymbol{\kappa}^{(i+1)} = \boldsymbol{\kappa}^{(i)} + \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}^{(i+1)}. \tag{18}$$

Numerical implementation of the augmented Lagrangian formulation for the projection of accelerations is given by the recursive set of equations

$$(\mathbf{M} + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha} \boldsymbol{\Phi}_q)\ddot{\mathbf{q}}^{(i+1)} = \mathbf{M}\ddot{\mathbf{q}}^{(i)} - \boldsymbol{\Phi}_q^T \boldsymbol{\alpha}(\dot{\boldsymbol{\Phi}}_q \dot{\mathbf{q}} + \boldsymbol{\Phi}_t) \tag{19}$$

and the recursion is started by setting

$$\ddot{\mathbf{q}}^{(0)} = \ddot{\mathbf{q}}^* \quad \text{and} \quad \boldsymbol{\kappa}^{(0)} = \boldsymbol{\alpha} \ddot{\boldsymbol{\Phi}}^{(0)}. \tag{20}$$

Equation (19) constitutes a system of linear algebraic equations in $\ddot{\mathbf{q}}$ that are solved iteratively until convergence in the generalized accelerations is achieved. Since the leading matrix in (19) is the same as that of Equation (13) used for the velocity projection, the acceleration analysis will require only successive forward reductions and back-substitutions.

2.4. Modified State-Space Formulation

A state-space representation of the equations of motion can be obtained using the basis of the null-space of the Jacobian. Such basis is defined by the matrix $\mathbf{R}$ which will satisfy the following condition [1]:

$$\Phi_q \mathbf{R} = 0. \tag{21}$$

Accordingly, the dependent velocities and accelerations can be expressed in terms of the independent ones as follows:

$$\dot{\mathbf{q}} = \mathbf{R}\dot{\mathbf{z}} \quad \text{and} \quad \ddot{\mathbf{q}} = \mathbf{R}\ddot{\mathbf{z}} + \dot{\mathbf{R}}\dot{\mathbf{z}}. \tag{22}$$

Introducing Equation (22) into Equation (3) the following equation is easily obtained:

$$\mathbf{R}^T\mathbf{M}\mathbf{R}\ddot{\mathbf{z}} = \mathbf{R}^T\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{R}^T\mathbf{M}\dot{\mathbf{R}}\dot{\mathbf{z}}, \tag{23}$$

which constitutes the equations of motion in independent co-ordinates. Equation (23) provides the function evaluation for the numerical integration scheme. This method requires the solution of the position and velocity problems [1] at each time step to obtain the matrix $\mathbf{R}$, $\mathbf{q}$ and $\dot{\mathbf{q}}$, so that Equation (23) may be solved. The solution of the position and velocity problems, that are costly computationally, have the same meaning as that of projecting the solution to obtain the dependent positions and velocities.

Implicit integrators make successive calls to (23) as a fixed point iteration. Such iteration is either generally inefficient or precludes convergence for stiff systems. Due to this reason we propose below a modification of this method based on the direct combination of Equation (23) with the difference equations of the numerical integration scheme. Such combination permits a modified Newton–Raphson iteration that allows this method to be used in stiff problems, and is described in detail below.

2.5. Fully-Recursive Formulation

A topological, fully-recursive algorithm of order $O(N)$ has been presented by Jiménez [24]. This algorithm uses a set of relative co-ordinates $\mathbf{z}$ to represent the state of the mechanism and the Cartesian velocities

$$\mathbf{Z}_i^t = \{\dot{\mathbf{r}}_{0i}^t \quad \omega_i^t\} \tag{24}$$

and their derivatives $\dot{\mathbf{Z}}_i$ to formulate the equations of motion. The vector $\dot{\mathbf{r}}_{0i}$ represents the velocity of the point of body i that is currently situated at the origin of the inertial reference frame, while ω_i is the angular velocity of body i. In the case of open chain mechanisms, the relative co-ordinates $\mathbf{z}$ are independent, equal in number to the degrees of freedom. If there are close chains, they are opened either at a pair or body, and the corresponding constraints are established.

Once the co-ordinates are defined, the first step consists of a kinematic transmission starting from the base towards the end of the chain. The corresponding recursive expressions in velocities and accelerations are given by

$$\mathbf{Z}_i = \mathbf{Z}_{i-1} + \mathbf{b}_i \dot{\mathbf{z}}_i, \tag{25}$$

$$\dot{\mathbf{Z}}_i = \dot{\mathbf{Z}}_{i-1} + \mathbf{b}_i \ddot{\mathbf{z}}_i + \mathbf{d}_i, \tag{26}$$

where $\mathbf{b}_i$ is a 6×1 vector containing the value of the Cartesian velocities $\mathbf{Z}_i$ when all the velocities $\dot{\mathbf{z}}$ are made zero except $\dot{\mathbf{z}}_i = 1$, and $\mathbf{d}_i$ is also a 6×1 vector containing the difference of the Cartesian accelerations $\dot{\mathbf{Z}}_i - \dot{\mathbf{Z}}_{i-1}$, when all the accelerations $\ddot{\mathbf{z}}$ are made zero.

The second step is a condensation of the forces and inertial terms starting from the end of the chain and progressing towards the base. In cases with closed chains, the reaction forces coming from points where chains have been opened are introduced through a penalty formulation as functions of the violation of the constraints and their derivatives. These reaction forces are condensed together with the rest of the forces. The recursive expressions for this second step are

$$\hat{\mathbf{M}}_{i-1} = \mathbf{M}_{i-1} + \mathbf{K}_i \hat{\mathbf{M}}_i, \tag{27}$$

$$\hat{\mathbf{Q}}_{i-1} = \mathbf{Q}_{i-1} + \mathbf{K}_i (\hat{\mathbf{Q}}_i - \hat{\mathbf{M}}_i \mathbf{d}_i), \tag{28}$$

where

$$\mathbf{K}_i = \mathbf{I}_6 - \frac{\hat{\mathbf{M}}_i \mathbf{b}_i \mathbf{b}_i^t}{\mathbf{b}_i^t \hat{\mathbf{M}}_i \mathbf{b}_i}. \tag{29}$$

$\mathbf{M}_i$ and $\mathbf{Q}_i$ are, respectively, the mass matrix and the vector of generalized forces of body i.

The third and last step is the calculation of accelerations $\ddot{\mathbf{z}}$ from the base to the end of the chain. For this purpose, the following expressions are provided

$$\ddot{\mathbf{z}}_i = \frac{\mathbf{b}_1^t (\hat{\mathbf{Q}}_i - \hat{\mathbf{M}}_1 \mathbf{d}_1)}{\mathbf{b}_1^t \hat{\mathbf{M}}_1 \mathbf{b}_1}, \tag{30}$$

$$\ddot{\mathbf{z}}_i = \frac{\mathbf{b}_i^t [\hat{\mathbf{Q}}_i - \hat{\mathbf{M}}_i (\mathbf{d}_i + \dot{\mathbf{Z}}_{i-1})]}{\mathbf{b}_i^t \hat{\mathbf{M}}_i \mathbf{b}_i}. \tag{31}$$

Therefore, it may be seen that the formulation is fully recursive and there is no need of solving a system of equations. This fact suggests a good speed competitiveness of this method as the size of the problem increases. In those cases, non-recursive formulations are forced to cope with large sets of equations. On the other hand the complexity of this formulation and its implementation is far superior to those seen above, especially the index-1 and index-3 methods.

3. Numerical Implementation of the Different Methods

The difference equations of the numerical integration scheme may be expressed in general terms as:

$$\dot{\mathbf{q}}_{n+1} = a\mathbf{q}_{n+1} - \hat{\dot{\mathbf{q}}}_n \quad \text{and} \quad \ddot{\mathbf{q}}_{n+1} = b\mathbf{q}_{n+1} - \hat{\ddot{\mathbf{q}}}_n, \tag{32}$$

where a and b are constants that depend on the numerical integration scheme and the time step, $\hat{\dot{\mathbf{q}}}_n$ and $\hat{\ddot{\mathbf{q}}}_n$ are known quantities that depend on the positions, velocities and accelerations at step n and/or previous steps. Note that if the method is explicit, a and b will be equal to zero.

3.1. Index-1 Augmented Lagrangian Formulation

Substituting Equation (32) in the equation of motion (5) for index-1, yields:

$$\begin{aligned}(\mathbf{M} + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha} \boldsymbol{\Phi}_q) b\mathbf{q}_{n+1} + \boldsymbol{\Phi}_q^T \boldsymbol{\lambda}^* + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha}(\dot{\boldsymbol{\Phi}}_q \dot{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_t) \\ - \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) - (\mathbf{M} + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha} \boldsymbol{\Phi}_q) \hat{\ddot{\mathbf{q}}} = 0,\end{aligned} \tag{33}$$

which constitutes a set of n equations of the form $\mathbf{f} = \mathbf{0}$ to be solved for the unknown position $\mathbf{q}_{n+1}$. We can apply a modified Newton–Raphson method and evaluate the solution by means of the iterative process:

$$\left[\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right]^{(i)} \Delta \mathbf{q}^{(i+1)} = -\mathbf{f}(\mathbf{q}^{(i)}), \tag{34}$$

where the function $\mathbf{f}$ is defined as

$$\mathbf{f}(\mathbf{q}_{n+1}) = \mathbf{M}\ddot{\mathbf{q}}_{n+1} + \boldsymbol{\Phi}_q^T(\boldsymbol{\alpha}\ddot{\boldsymbol{\Phi}} + \boldsymbol{\lambda}^*) - \mathbf{Q}_{n+1}, \tag{35}$$

and the tangent matrix is approximated by the following matrix:

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}} \cong b(\mathbf{M} + \boldsymbol{\Phi}_q^T \boldsymbol{\alpha} \boldsymbol{\Phi}_q) - \mathbf{Q}_q - a\mathbf{Q}_{\dot{\mathbf{q}}}, \tag{36}$$

where the last two terms of (36) represent the contribution to the tangent matrix coming from the elastic forces (i.e. springs) and those dependent on the velocity (i.e. dashpots).

3.2. INDEX-3 AUGMENTED LAGRANGIAN FORMULATION

The substitution of Equation (32) into Equation (15) for index-3 yields

$$b\mathbf{M}\mathbf{q}_{n+1} + \boldsymbol{\Phi}_q^T(\alpha\boldsymbol{\Phi} + \boldsymbol{\lambda}^*) = \mathbf{Q}_{n+1} + \mathbf{M}\hat{\ddot{\mathbf{q}}}_n, \tag{37}$$

which again constitutes a set of nonlinear algebraic equations with $\mathbf{q}_{n+1}$ as unknowns. We use a modified Newton–Raphson iteration and evaluate the solution by means of the iterative process outlined above with the function $\mathbf{f}$ defined as

$$\mathbf{f}(\mathbf{q}_{n+1}) = \mathbf{M}\ddot{\mathbf{q}}_{n+1} + \boldsymbol{\Phi}_q^T(\alpha\boldsymbol{\Phi} + \boldsymbol{\lambda}^*) - \mathbf{Q}_{n+1} \tag{38}$$

and the tangent matrix approximated by

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}} \cong b\mathbf{M} + \boldsymbol{\Phi}_q^T \alpha \boldsymbol{\Phi}_q - \mathbf{Q}_q - a\mathbf{Q}_{\dot{\mathbf{q}}}. \tag{39}$$

3.3. NEW STATE-SPACE IMPLEMENTATION FOR STIFF AND NON-STIFF SYSTEMS

One of the major drawbacks of the state-space representation given by Equation (23) is that this equation is customarily used as a fixed point iteration for function evaluation. It is well known in numerical analysis that such iteration is not efficient and even fails when the equations of motion are stiff. Consequently, in what follows, we propose a combination of the equations of motion (23) and the finite difference equations of the numerical integration scheme (32), such that the integration of both stiff and non-stiff problems can be made tractable. Since we are now integrating using independent co-ordinates, Equation (32) may be rewritten as:

$$\dot{\mathbf{z}}_{n+1} = a\mathbf{z}_{n+1} - \hat{\dot{\mathbf{z}}}_n \quad \text{and} \quad \ddot{\mathbf{z}}_{n+1} = b\mathbf{z}_{n+1} - \hat{\ddot{\mathbf{z}}}_n. \tag{40}$$

The substitution of Equation (40) in the equation of motion for state-space (23), yields:

$$\mathbf{R}^T\mathbf{M}\mathbf{R}b\mathbf{z}_{n+1} - \mathbf{R}^T\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{R}^T\mathbf{M}\dot{\mathbf{R}}a\mathbf{z}_{n+1} - \mathbf{R}^T\mathbf{M}\mathbf{R}\hat{\ddot{\mathbf{z}}} - \mathbf{R}^T\mathbf{M}\dot{\mathbf{R}}\hat{\dot{\mathbf{z}}} = 0, \tag{41}$$

which constitutes a set of n equations of the form $\mathbf{f} = \mathbf{0}$ to be solved for the unknowns $\mathbf{z}_{n+1}$. Again, we can apply a modified Newton–Raphson method and evaluate the solution by means of the iterative process:

$$\left[\frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right]^{(i)} \Delta\mathbf{z}^{(i+1)} = -\mathbf{f}(\mathbf{z}^{(i)}), \tag{42}$$

where the function $\mathbf{f}$ is defined as:

$$\mathbf{f}(\mathbf{z}_{n+1}) = \mathbf{R}^T\mathbf{M}\mathbf{R}\ddot{\mathbf{z}}_{n+1} - \mathbf{R}^T\mathbf{Q}_{n+1} + \mathbf{R}^T\mathbf{M}\dot{\mathbf{R}}\dot{\mathbf{z}}_{n+1} = 0, \tag{43}$$

and the tangent matrix is approximated by the following matrix:

$$\frac{\partial \mathbf{f}}{\partial \mathbf{z}} \cong b(\mathbf{R}^T\mathbf{M}\mathbf{R}) - \mathbf{R}^T\mathbf{Q}_q\mathbf{R} - a\mathbf{R}^T\mathbf{Q}_{\dot{\mathbf{q}}}\mathbf{R}, \tag{44}$$

Table I. Comparative results for the double pendulum.

	$\Delta t = 10^{-2}$		$\Delta t = 10^{-3}$		$\Delta t = 10^{-4}$		$\Delta t = 10^{-5}$	
	Error	Time	Error	Time	Error	Time	Error	Time
Index-1	No convergent.		1.1E-3	2.8	1.1E-5	21.4	2.5E-7	214
Index-3	2.1E-1	0.4	4.6E-3	1.9	2.0E-5	17.2	Wrong results	
State-space	2.9E-1	1.8	4.8E-3	10.5	4.3E-5	96.6	7.6E-7	862
Fully-rec	2.7E-1	2.0	2.6E-3	11.3	3.4E-5	100.3	3.4E-7	1003

where the last two terms of (44) represent the contribution to the quasi-tangent matrix coming from the elastic forces (i.e. springs) and forces dependent on the velocity (i.e. dashpots). The contribution of elastic and damping terms to the tangent matrix makes this method more suitable and efficient for stiff systems than the classical fixed point iteration which only includes the first term in the r.h.s. of (44).

3.4. Fully-Recursive Formulation

As it may be easily understood after the description of this formulation given in the previous section, the own nature of the fully recursive method makes difficult to introduce the integrator scheme (40) into the dynamic equations, since these equations are not explicitly formulated, but developed in three different recursive steps. Therefore, in this case the integration of the equations of motion must be performed using a fixed point iteration. The corresponding function evaluations (acceleration analysis) are provided by the described recursive formulation, and the integration scheme (40) changes its form to

$$\mathbf{z}_{n+1} = \frac{1}{b}\,\ddot{\mathbf{z}}_{n+1} - \hat{\mathbf{z}}_n \quad \text{and} \quad \dot{\mathbf{z}}_{n+1} = \frac{1}{a}\,\ddot{\mathbf{z}}_{n+1} - \hat{\dot{\mathbf{z}}}_n. \tag{45}$$

This fact constitutes a drawback when comparing this method to the other formulations described above which apply a Newton–Raphson scheme. However, as said before, this method does not require the solution of any set of linear equations. We will show below which effects dominate in different cases.

4. Numerical Results

First Example. The simulation of a double pendulum that moves from rest in the horizontal position under gravity effects is carried out. Each link has a distributed unit mass and a unit length. We use natural co-ordinates for the modeling process [1] and for the integration we use the trapezoidal rule which is second order, implicit, A-stable method and also energy preserving in the linear regime. The total time of simulation is 10 seconds and the penalty factor is 10^7.

Table I shows the maximum error in the energy norm and the CPU time in seconds taken by each of the four methods for different time steps of integration.

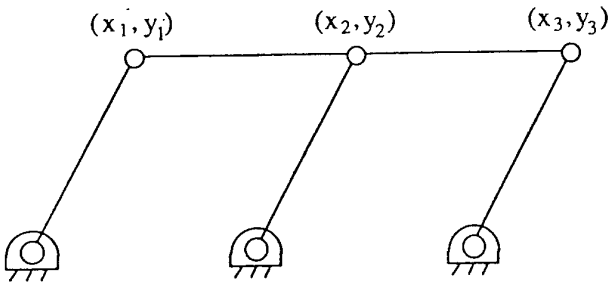


Figure 2. Assembly of two four-bar linkages.

It is important to emphasize that the index-3 formulation does not converge for the time steps 10^{-4} and 10^{-5} unless the penalty factor is increased to 10^9 and 10^{11}, respectively. Even in the last case the results obtained are wrong due to the ill-conditioning caused by the small time steps in the index-3 formulation (the reader is referred to [17] for an explanation of this effect). Note, however, that no pivoting or filtering is done in the solution process to avoid or diminish this ill-conditioning.

The index-1 method is more robust than the index-3. It does not require to modify the penalty factor and increases its accuracy as the time step is decreased, as clearly seen in the table. The index-3 method is more efficient than the index-1 for large time steps of integration.

The state-space and the fully recursive methods work for all the time steps considered. Therefore, they show a robust behavior although they run notably slower than its two competitors. In this example, the slowness of the fourth method is justified due to the reduced number of variables used by the other methods (just 4). Moreover, the fully-recursive formulation has been developed for three-dimensional cases and 6×6 matrices and 6×1 vectors are used. Undoubtedly, this formulation could be simplified for planar cases, thus getting smaller calculation times.

Second Example. The simulation of a one degree-of-freedom assembly of two four-bar linkages has been proposed in [25] as an example to test the performance in cases where the mechanism undergoes singular configurations. When the mechanism reaches a horizontal position, the number of degrees of freedom increases instantaneously from 1 to 3. To define the position of the system, we use the six position variables (x_1, y_1, x_2, y_2, x_3, y_3), as may be seen in Figure 2. All the links are of length $l = 1$ m and have a uniformly distributed mass $m = 1$ kg. The gravity force acts in the negative Y direction, with a value $g = 9.81$ m/s^2. At $t = 0$ the initial velocity is $\dot{x}_1 = 1$. We integrate the motion for 10 seconds with 10^7 the value of the penalty parameter. Figure 3 shows different time-histories of the simulation and shows that the mechanism goes through the singular position ($|x_1| = 1$) 10 times.

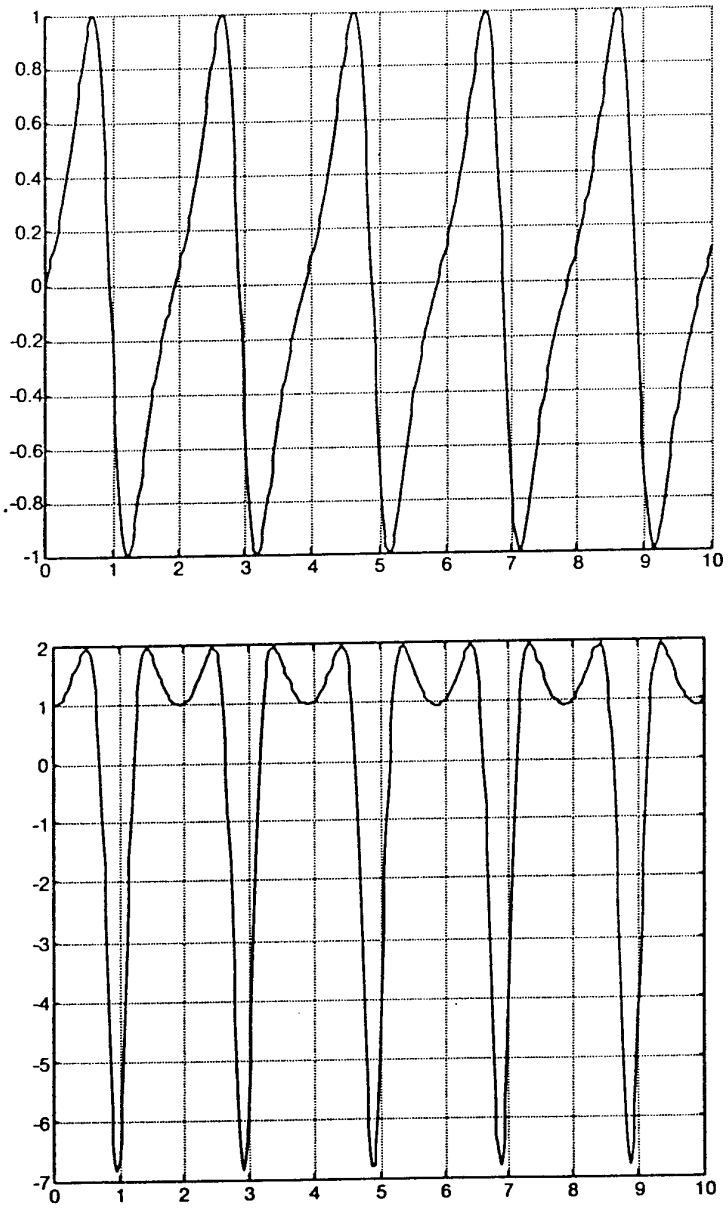


Figure 3. Time-history of the two four-bar linkages.

The results obtained are illustrated in Table II. The same trend observed in the previous case occurs again in this one. The index-1 method is more robust in the sense that it increases its accuracy as Δt decreases although it does not converge for big time steps. The index-3 is more efficient and converges for bigger time steps but shows moderate errors for 10^{-3} and fails for time steps equal to or smaller than 10^{-4}. As expected [25], neither the state-space method nor the fully-recursive

Table II. Comparative results for the two four-bar linkages.

	$\Delta t = 5 * 10^{-2}$		$\Delta t = 10^{-2}$		$\Delta t = 10^{-3}$		$\Delta t = 10^{-4}$	
	Error	Time	Error	Time	Error	Time	Error	Time
Index-1	No convergent.		5.6E-3	1.0	3.0E-4	7.2	2.8E-7	63.6
Index-3	7.6E-2	0.3	5.0E-3	0.6	7.7E-2	4.0	Wrong results	
State-space	Does not work		Does not work		Does not work		Does not work	
Fully-rec	Does not work		Does not work		Does not work		Does not work	

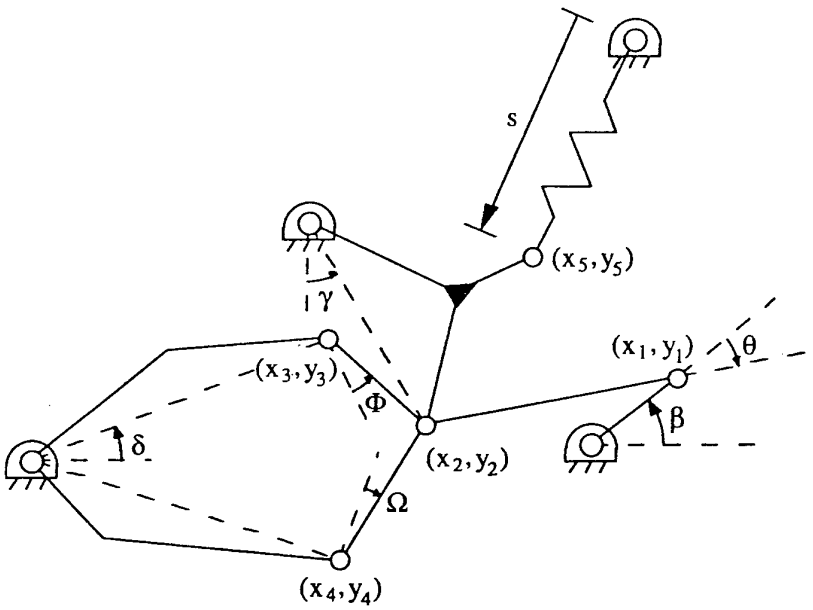


Figure 4. Andrew's mechanism.

formulation work in these kind of simulations because they either fail or lead to divergent results when the singular position is reached.

Third Example. The Andrew's squeezing mechanism, described in detail in [26], has been proposed as an example to test numerical algorithms in multibody dynamics. It requires very small time steps because of the small time scale of the problem. We perform the simulation using natural co-ordinates and redundant constraints to introduce the angle of the crank β as an additional variable (see Figure 4). The penalty factor is $\alpha = 10^7$, and again the trapezoidal rule is used for the numerical integration for a total time of 0.03 seconds. Sparse solvers are used in the non-recursive methods to solve the system of equations. The simulation yields the values of the angles shown in Figure 5, which are calculated in a post-process as functions of the natural co-ordinates used to model the mechanism. The angles in Figure 4 and the plot numbers in Figure 5 are related as follows: $1-\beta, 2-\theta, 3-\gamma, 4-\delta, 5-\Phi, 6-\Omega$. The analysis is performed using the four proposed methods and

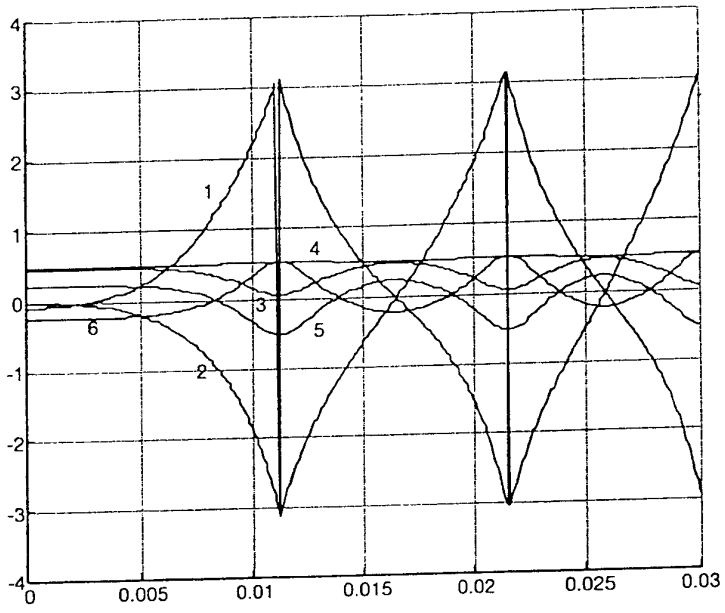


Figure 5. Angles of the Andrew's mechanism.

Table III. Comparative results for the Andrew's mechanism.

	$\Delta t = 10^{-4}$		$\Delta t = 10^{-5}$		$\Delta t = 10^{-6}$	
	Error	Time	Error	Time	Error	Time
Index-1	1.4E-2	0.6	2.5E-2	5.2	2.0E-4	54
Index-3	Wrong results		No convergent.		No convergent.	
State-space	4.3E-3	1.3	9.3E-5	10.2	6.2E-5	101.9
Fully-rec	3.0E-3	2.8	9.3E-5	16.2	3.2E-5	145.6

the results are depicted in Table III. It may be seen that since the time step required is very small, the index-3 method either fails or yields wrong results (again no pivoting or filtering is performed). On the other hand, the index-1 method yields good results and shows better accuracy as the time step decreases. The state-space method shows this behavior also, giving even a slightly better energy conservation than index-1, but running at half of the speed. The fully-recursive formulation presents also a robust behavior, although it runs the slowest of all because of the number of close kinematic loops that the problem has. The system of equations for the index-1 and index-3 methods are solved using solvers that take advantage of the sparse nature of their leading matrices.

Fourth Example. The 4 × 4 Iltis vehicle [27] has been proposed as a benchmark problem by the European automobile industry to check multibody dynamic codes. We perform two different simulations using the front suspension of the vehicle (see Figure 6). The model requires a total of 23 variables, related through 22 constraint

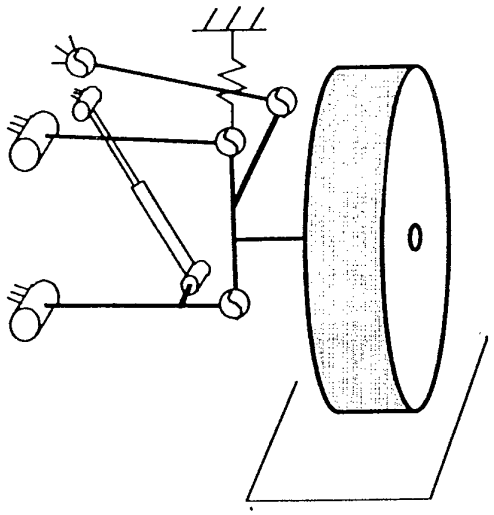


Figure 6. Iltis suspension.

equations with only one degree of freedom. The characteristics of the suspension are:

- A *shock absorber*, which provides a damping force given by the following nonlinear expression:

$$F_D = 9945.627v + 33955.72v^2 - 59832.25v^3 - 395651.0v^4 \text{ [N]}$$

$$\text{for } -0.2 < v < 0.21 \text{ m/s}$$

$$F_D = -416.42 + 1844.3v \text{ [N]} \quad \text{for } v < -0.2 \text{ m/s}$$

$$F_D = 1919.1638 + 1634.727v \text{ [N]} \quad \text{for } v < 0.21 \text{ m/s}$$

and a elastic force due to an external polymer, given by

$$F_S = -4.0092 * 10^6 + 2.8397 * 10^7 x - 6.7061 * 10^7 x^2 + 5.2796 * 10^7 x^3 \text{ [N]},$$

where the distance x is defined in meters.

- A *tire*, modeled by means of a linear vertical spring of stiffness 460,000 N/m.
- A *leaf spring*, modeled as a linear spring of 35,900 N/m for deformations smaller than 14.5 cm. When such deformation is exceeded the suspension hits a second spring of value 10^7 N/m, thus adding a much higher stiffness to the suspension.

The first simulation is performed as follows: the suspension starts moving from at rest conditions at a speed of 5 m/s, and its position does not correspond to the static equilibrium. Thus, it freely oscillates until the static equilibrium position is reached. After three seconds, the suspension goes over a road bump, defined as a

Table IV. Comparative results for the Iltis front suspension.

	$\Delta t = 10^{-2}$		$\Delta t = 10^{-3}$		$\Delta t = 10^{-4}$		$\Delta t = 10^{-5}$	
	Error	Time	Error	Time	Error	Time	Error	Time
Index-1	No convergent.		1.2E-3	101.2	6.0E-5	464.9	3.0E-7	4791
Index-3	1.6E-1	6.0	2.1E-3	38.3	7.5E-2	377.2	Wrong results	
State-space	1.6E-1	16.7	7.0E-4	137.0	4.0E-5	1091.0	4.0E-7	10435
Fully-rec	6.6E-2	6.6	4.2E-4	38.6	6.0E-6	392.0	2.0E-7	4079

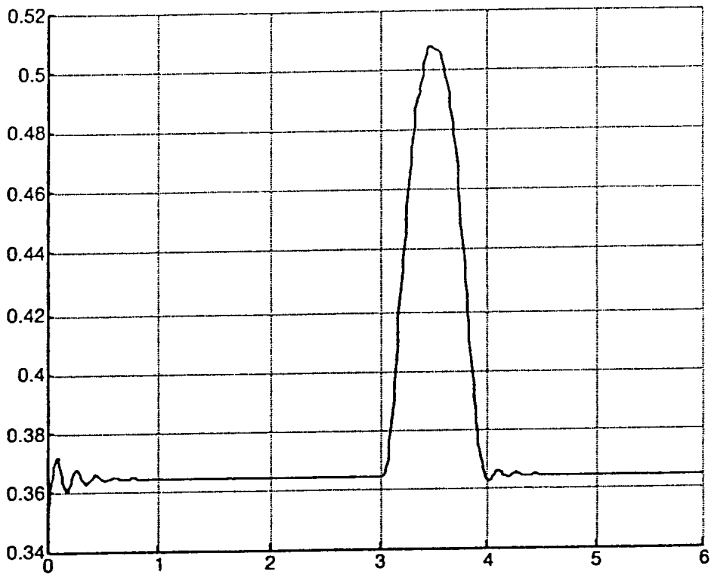


Figure 7. Vertical position of the wheel center. Non-stiff case.

cosine function, and afterwards oscillates until the equilibrium is reached again. The stiff spring of value 10^7 N/m is not activated in this simulation, therefore, it cannot be considered as a stiff problem. The complete analysis lasts for 6 seconds. The analysis is carried out using the four methods and the performance results are depicted in Table IV. Figure 7 illustrates the time history of the center of the wheel.

Table IV shows that the index-3 method does not converge for time steps 10^{-4} and 10^{-5} unless the penalty factor is increased to 10^{11} and 10^{13}, respectively. For a step of 10^{-5} the results obtained are wrong because of the ill-conditioning. The index-1 method is again much more robust, not requiring to modify the penalty factor, whose value is set to 10^7 for all the time steps. Its accuracy increases as the time step is decreased, but the method does not converge for $\Delta t = 10^{-2}$ (no pivoting is used). The state-space method also shows a robust behavior although, in this case, it runs notably slower than the other methods.

Table IV also shows that the fully-recursive formulation becomes competitive in the simulation of this non-stiff large system. The reason for this behavior is that in

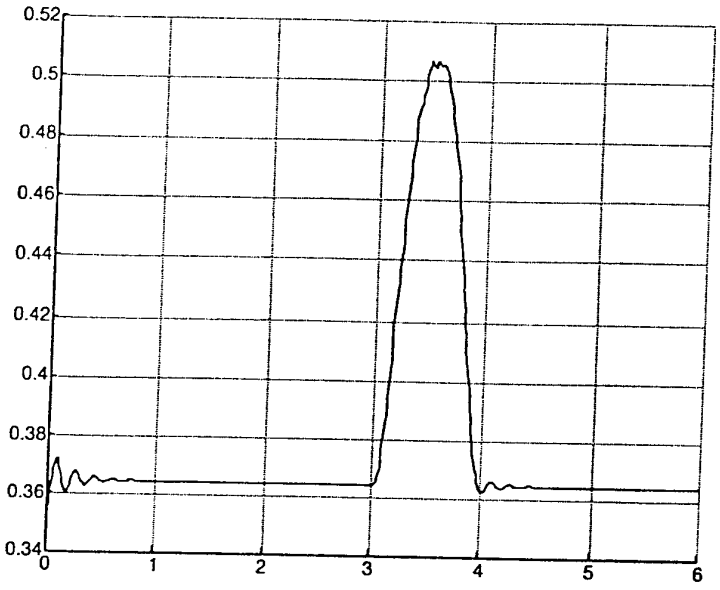


Figure 8. Vertical position of the wheel center. Stiff case.

this example the first non-recursive methods use 23 variables, notably more than in the previous examples, and require the triangularization of matrices of size $23 * 23$ (sparse matrix algebra has been used to solve these equations). Conversely, the fully-recursive method does not require the solution of such system of equations in each function evaluation. It is important to note how the index-3 method and the fully-recursive method take very similar times of integration, thus corroborating the equivalence between exploiting sparsity and the articulated body approach for systems with a large number of bodies.

The second simulation is performed activating the second spring so that when the suspension hits the bump and the leaf-spring exceeds a deformation of 14.5 cm the spring of value 10^7 N/m becomes active, thus introducing additional stiffness to the system. The simulation is performed using the index-1 and the fully recursive formulations at different time steps of integration. Figure 8 shows the time history of the center of the wheel for this case. It is important to point out that the fully-recursive method requires a much smaller time step, $5.0*10^{-6}$ seconds compared to the index-1 method which successfully performs the simulation starting at $5.0*10^{-5}$ seconds. It becomes obvious that for stiffer systems such as those resulting from larger springs, visco-elastic bushings, flexible multibodies, active control systems, etc., the performance of the fully recursive method will further degrade.

As a result of this example, it is clear that a multi-index-1-3 approach with variable time step would provide an ideal solution, since the more efficient index-3 method would be used during a large part of the motion and the index-1 only at that part where the time step decreases below 10^{-4}.

5. Conclusions

We have addressed in this paper the issue of how the modeling aspects of dependent *versus* independent co-ordinates, descriptor form *versus* state-space form of the equations of motion, and global *versus* topological methods affect the dynamic simulation of multibody systems. More concretely, we have examined and compared the use of the top of the line methods that include: an augmented Lagrangian formulation with projections in *index-1* and *index-3*, a new *modified state-space* formulation based on projection matrices, that is capable of handling the numerical stiffness coming from built in stiff numerical properties (springs, dashpots, etc.), and a *fully-recursive* formulation whose calculation effort grows linearly with the size of the problem. The aim is to assess how these factors may be taken into account to establish a new concept in intelligent simulation.

These are the major conclusions:

- The index-3 formulation with projections is the most efficient of all the methods tested, but it leads to serious ill-conditioning at small time steps, starting at about 10^{-5}, when the penalty parameter becomes problem dependent. (Note that neither pivoting nor filtering techniques have been used to avoid or diminish ill-conditioning.)
- The index-1 formulation with projections is the most robust of all the methods, but it cannot converge for large time steps where the index-3 does. Contrary to index-3 it becomes more accurate as the time step decreases and it is not affected by ill-conditioning. It is also insensitive to the penalty parameter which remained constant in all the simulations.
- The new state-space method works well with stiffness, shows robust behavior and good energy preservation, as well as good convergence at all time steps. However, it runs considerably slower than the index-1 and index-3 methods in all the cases tested.
- The fully-recursive (articulated inertia) formulation behaves in a similar way to that of the previous one, except in what refers to speed and stiffness. This method is much faster than the state-space but not suitable for stiff problems (such as those resulting from large springs, visco-elastic bushings, flexible multibodies, active control systems, etc.). Also, it turns out that for small systems its performance is inferior to the index-3 and index-1 methods, but improves as the size of the problem increases. For large non-stiff problems the method becomes competitive.
- The following tables summarize the findings of this paper. Each method is given a grade depending on its performance under different situations. The letter "A" means outstanding, "B" good, "C" fair, "D" poor and "F" signifies a failure.

Method	Easiness of implementation	Free from graph methods	Speed		Accuracy		
			Small to medium problems	Medium to large problems	$\Delta t < 10^{-5}$	$\Delta t > 10^{-5}$ $\Delta t < 10^{-2}$	$\Delta t > 10^{-2}$
Index-1	A	A	B	B	A	A	F
Index-3	A	A	A	A	F	A	A
Space-state	B	A	C	C	A	A	A
Recursive	C	D	C	A	A	A	A

Method	Performance				
	Changing topologies	Singular configurations	Inequality constraints	Numerical stiffness	Redundant constraints
Index-1	A	A	A	A	A
Index-3	A	A	A	A	A
Space-state	C	F	C	A	C
Recursive	D	F	D	D	D

- It remains to address in future work the real-time behavior of these methods on large scale industrial problems, such as the complete model of an automobile. More concretely, a multi-index-1/index-3 formulation could lead to an ideal situation since it encompasses the benefits of both methods [28]. Also sparse-parallel solvers have a lot of potential for the index-1/index-3 approach since a great deal of operations may be performed element-by-element. On the other hand, the fully recursive formulation becomes competitive on large non-stiff problems, although to be considered as a general tool it will require graph methods for identifying close-chain loops and spanning trees. Therefore, the recursive method is a good candidate for special purpose simulators of large non-stiff multibody systems.

Acknowledgments

The support of this work provided by the United States Army Research Office under grant DAAH04-95-1-0662, and the CICYT of the Spanish Ministry of Education under grant TAP95-0226 is gratefully acknowledged.

References

1. García de Jalón, J. and Bayo, E., *Kinematic and Dynamic Simulation of Multibody Systems*, Springer-Verlag, Berlin, 1994.
2. Baumgarte, J., 'Stabilization of constraints and integrals of motion in dynamical systems', *Computer Methods in Applied Mechanics and Engineering* **1**, 1972, 1–16.
3. Bayo, E. and Ledesma, R., 'Augmented Lagrangian and mass-orthogonal projection methods for constrained multibody dynamics', *Nonlinear Dynamics* **9**, 1996, 113–130.
4. Kurdila, A.J., Junkins, J.L. and Hsu, S., 'Lyapunov stable penalty methods for imposing non-holonomic constraints in multibody system dynamics', *Nonlinear Dynamics* **4**, 1993, 51–82.

5. Wehage, R.A. and Haug, E.J., 'Generalized co-ordinate partitioning for dimension reduction in analysis of constrained dynamic systems', *ASME Journal of Mechanical Design* **104**, 1982, 247–255.
6. Kane, T.R. and Levinson, D.A., *Dynamics: Theory and Applications*, McGraw-Hill, New York, 1985.
7. Serna, M.A., Avilés, R. and García de Jalón, J., 'Dynamic analysis of planar mechanisms with lower-pairs in basic co-ordinates', *Mechanism and Machine Theory* **17**, 1982, 397–403.
8. Jerkovsky, W., 'The structure of multibody dynamic equations', *Journal of Guidance and Control* **1**, 1978, 173–182.
9. Kim, S.S. and Vanderploeg, M.J., 'A general and efficient method for dynamic analysis of mechanical systems using velocity transformations', *Journal of Mechanisms, Transmissions and Automation in Design* **108**, 1986, 176–182.
10. Nikravesh, P.E. and Gim, G., 'Systematic construction of the equations of motion for multibody systems containing closed kinematic loops', *Advances in Design Automation* **3**, 1989, 27–33.
11. García de Jalón, J., Avello, A. Jiménez, J.M., Martín, F. and Cuadrado, J., 'Real-time simulation of complex 3-D multibody systems with realistic graphics', in *Real-Time Integration Methods for Mechanical System Simulation*, E.J. Haug and R.C. Deyo (eds), NATO ASI Series, Vol. 69, Springer-Verlag, Berlin, 1990, 265–292.
12. Bae, D.S. and Won, Y.S., 'A Hamiltonian equation of motion for real time vehicle simulation', *Advances in Design Automation* **2**, 1990, 151–157.
13. Avello, A., Jiménez, J.M., Bayo, E. and García de Jalón, J., 'A simple and highly parallelizable method for real-time dynamic simulation based on velocity transformations', *Computer Methods in Applied Mechanics and Engineering* **107**, 1993, 313–339.
14. Walker, M.W. and Orin, D.E., 'Efficient dynamic computer simulation of robotic mechanisms', *ASME Journal of Dynamic Systems Measurement and Control* **104**, 1982, 205–211.
15. Featherstone, R., 'The calculation of robot dynamics using articulated body inertias', *The International Journal of Robotic Research* **2**, 1983, 13–30.
16. Featherstone, R., *Robot Dynamics Algorithms*, Kluwer Academic Publishers, Dordrecht, 1987.
17. Brenan, K.E., Campbell, S.L. and Petzold, L.R., *The Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, North-Holland, New York, 1989, 210 pp.
18. Griepentrog, E., Hanke, M. and Marz, R., *Berlin Seminar on Differential Algebraic Equations*, Fachbereich Mathematik der Humboldt-Universität zu Berlin, 1992.
19. Führer, C. and Leimkuhler, B.J., 'Numerical solution of differential-algebraic equations for constrained mechanical motion', *Numerische Mathematik* **59**, 1991, 55–69.
20. Lubich, Ch., Nowak, U., Pöhle, U. and Engstler, Ch., 'MEXX – Numerical software for the integration of constrained mechanical multibody systems', Preprint SC 92-12, Konrad-Zuse-Zentrum, Berlin, 1992.
21. Arnold, M., 'Stability of numerical methods for differential-algebraic equations of higher index', *Applied Numerical Mathematics* **13**, 1993, 5–14.
22. Hairer, E. and Wanner, G., *Solving Ordinary Differential Equations II*, Springer-Verlag, Berlin, 1996.
23. Petzold, L.R., 'A description of DASSL: A differential/algebraic system solver', in *IMACS Transactions of Scientific Computation*, Vol. 1, R.S. Stepleman (ed.), Montreal, Canada, 1982.
24. Jiménez, J.M., 'Kinematic and dynamic formulations for real-time simulation of multibody systems', Ph.D. Thesis, University of Navarra, Spain, 1993.
25. Bayo, E. and Avello, A., 'Singularity free augmented Lagrangian algorithms for constraint multibody dynamics', *Nonlinear Dynamics* **5**, 1994, 209–231.
26. Schiehlen, W.O., *Multibody System Handbook*, Springer-Verlag, Heidelberg, 1990.
27. Iltis Data Package, IAVSD Workshop, Herbertov, Czechoslovakia, September 1990.
28. Cardenal, J., Cuadrado, J. and Bayo, E., 'A multi-index variable time step method for the dynamic simulation of multibody systems', in *ASME Design Engineering Technical Conferences and Computers in Engineering Conference*, J.M. McCarthy (ed.). Irvine, CA, 1996, CD ROM, DETC/CIE-1625.

PUBLICATION 2

A MULTI-INDEX VARIABLE TIME STEP METHOD FOR THE DYNAMIC SIMULATION OF MULTIBODY SYSTEMS

J. CARDENAL

Department of Mathematical Methods in Engineering, University of La Coruña. 15403 Ferrol, Spain

J. CUADRADO AND P. MORER

Department of Mechanical Engineering, University of La Coruña. 15403 Ferrol, Spain

E. BAYO

Department of Mechanical Engineering, University of La Coruña and School of Engineering and Architecture, University of Navarra. 31080 Pamplona, Spain.

SUMMARY

This paper presents a multi-index variable time step method for the integration of the equations of motion of constrained multibody systems in *descriptor form*. The basis of the method is the augmented Lagrangian formulation with projections in index-3 and index-1. The method takes advantage of the better performance of the index-3 formulation for large time steps and of the stability of the index-1 for low time steps, and automatically switches from one method to the other depending on the required accuracy and values of the time step. Various numerical problems that arise during the simulation process are described. The paper also describes ways to circumvent problems.
The variable time stepping is accomplished through the use of an integral of motion, which in the case of conservative systems becomes the total energy. The error introduced by the numerical integrator in the integral of motion during consecutive time steps provides a good measure of the local integration error, and permits a simple and reliable strategy for varying the time step. It is also shown how the energy stored in the penalty system is suitable to measure the local integration error. Overall, the method is efficient and powerful; it is suitable for stiff and non-stiff systems, robust for all time step sizes, and it works for singular configurations, redundant constraints and topology changes. Also, the constraints in positions, velocities and accelerations are satisfied during the simulation process. The method is robust in the sense that it becomes more accurate as the time step size decreases.
A section is devoted at the end of the paper to present numerical simulations that illustrate the performance of the proposed method.

1. INTRODUCTION

Kinematics and dynamic simulations of multibody systems allow the accurate prediction of the behavior of heavy machinery, spacecraft, automobile suspensions and steering systems, graphic arts and textile machinery, robots, packaging machinery, machine tools, etc. The first issue to consider in the simulation process is that of modeling the system; that is, the selection of a set of parameters or coordinates that will allow to unequivocally define at all times the position, velocity and acceleration of the system. The most useful kinds of coordinates currently used to define the motion of multibody systems are relative coordinates, reference point (or Cartesian) coordinates, and natural (or fully Cartesian) coordinates. These coordinates, when combined with the principles of dynamics, lead to the final form of the equations of motion. Dynamic principles such as Lagrange's formulation, Newton's Laws, canonical equations of Hamilton, Virtual Power, Hamilton's Principle and Gibbs-Apell equations, constitute the basis for the formulations of multibody dynamics[1,2,3,4]. The choice of dynamic formulation determines the subsequent choice of numerical integration schemes.

The method of Lagrange's multipliers leads to a representation of the equations of motion in *descriptor form* constituting a set of index-3[†] differential algebraic equations (DAE). The addition of stabilization techniques, such as the method of Baumgarte[5], reduces the index and makes the solution tractable by means of standard ordinary differential equations (ODE) solvers, however, it does not provide full constraint satisfaction, leads to a limited control of accuracy, and in addition provides no way for choosing the values of the coefficients used by the method. An augmented Lagrangian formulation with projections[6] has been proposed which, in addition to transforming the set of equations into a stabilized set, is solvable by standard ODE methods and assure Lyapunov stability of the simulation process[7]. This method also has the advantages of being robust under singular configurations, topology changes and with redundant constraints, and provides full constraint satisfaction.

[†] Following the notation used by Brenan et al. (1989), given a DAE $F(t,y,y')=0$, we call *index* of that DAE the minimum number of times that all or part of the original DAE must be differentiated with respect to the independent variable (in this case t) in order to determine the derivative of the function, y', as a continuous function of y and t

State-space methods, such as coordinate partitioning[8], Kane's method[9] and virtual power with projection matrices[10], transform the equations of motion to a minimum set of coordinates that are directly solvable by ODE methods. State-space methods may also be obtained by means of *velocity transformations*[11,12,13,14,15]. State-space methods are more suitable for ODE integration than the descriptor counterparts at the expense of solving the velocity and position problem at each time step. However, they do not handle well either topology changes or singular configurations. In addition, obtaining the tangent matrix required for stiff integrators is extremely involved and practically unfeasible in analytical form using state-space methods. Consequently, they will not be a good choice when the system has numerical stiffness. Typically, numerical stiffness in multibody dynamics arises from the modeling of leaf springs, shock absorbers, bushings, contact and impact problems.

The numerical mathematics community has sought solutions to the index reduction problem and has proposed many different ways. Recent advances have been made which have yielded stable index reduction methods and accurate ways of projecting the DAE onto the underlying ODE for more stable and accurate solutions. Key developments are the work of Brenan, et al.[16], Griepentrog, et al.[17], Führer and Leimkuhler[18], Hairer and Wanner[19] (developers of RADAU5), Lubick[20,21] (developer of MEXX), Petzold[22] (developer of DASSL) and Arnold[23,24] (developer of HEX5).

In regard to numerical integration, the backward difference formula (BDF) methods have been customarily used for the solution of differential algebraic equations because the artificial damping thereby introduced helps to stabilize the solution and provides convergence particularly in the index-3 setting. However, the actual implementation of BDF algorithms in general-purpose solvers is not free from numerical difficulties, which become more acute for index-3 when the time step size is smaller than 10^{-4}–10^{-5} seconds. The main difficulties are:

- For an index-m DAE the tangent or quasi-tangent matrix used in the Newton-Raphson iteration has a condition number of order $O(1/\Delta t^m)$[16]. Hence, the practical implementation of the method is bound to have large round-off errors for small time steps (usually starting at $\Delta t=10^{-5}$).
- Instabilities may result from sudden changes in system variables and constraints, such as impacts, sudden appearances or disappearances of constraints and topology changes. Any time there is a discontinuity in the response the multi-step BDF tries to fit a polynomial through the discontinuity and therefore the time step size must be severely reduced. As explained in the previous point, this results in an ill-conditioned iteration matrix. Consequently the Newton-Raphson iteration may end up near a solution and yet not be able to converge to it. These problems can be circumvented, but at the expense of re-initializing the integration, thus producing delays in the integration process.

Due to the reasons suggested above we propose a multi-index, augmented Lagrangian formulation of the equations of motion in descriptor form for index-1 and index-3. The index-3 form is more efficient than the index-1; however, for time step sizes smaller than 10^{-5} the ill-conditioning of the tangent matrix in index-3 affects the performance of the method, while the index-1 form becomes more accurate and robust[25]. A single-step numerical integration scheme with variable index and time step size is developed based on the strategy mentioned above. Mass-orthogonal projections to the constraint space, as described in the work of Bayo and Ledesma[6], are performed to assure constraint satisfaction to machine precision during the integration process. At the end of the paper, numerical simulations are presented that illustrate the performance of the proposed approach.

2. PRELIMINARIES ON MULTI-BODY DYNAMIC ANALYSIS IN DESCRIPTOR FORM

2.1 Formulation in Fully Cartesian coordinates

Let us consider a multibody system whose configuration is characterized by n fully Cartesian (or natural) coordinates[2] denoted by vector $\mathbf{q}$ that are interrelated through the m holonomic kinematic constraint conditions:

$$\Phi(\mathbf{q},t)=0 \tag{1}$$

The use of the Principle of Virtual Power directly leads to the equations of motion:

$$\delta\mathbf{q}^T\left(\mathbf{M}\ddot{\mathbf{q}}-\mathbf{Q}(\mathbf{q},\dot{\mathbf{q}})+\Phi_{\mathbf{q}}^T\boldsymbol{\lambda}\right)=0 \tag{2}$$

which for a general multibody system leads to:

$$\mathbf{M}\ddot{\mathbf{q}}+\Phi_{\mathbf{q}}^T\boldsymbol{\lambda}=\mathbf{Q}(\mathbf{q},\dot{\mathbf{q}}) \tag{3}$$

where $\mathbf{M}$ is the mass matrix; $\mathbf{Q}$ is the vector of the external, conservative and non-conservative forces; Φ_q is the Jacobian of the constraint equations, and λ is a vector containing the Lagrange's multipliers. The use of fully Cartesian coordinates leads to a constant mass matrix $\mathbf{M}$, and the absence of velocity dependent inertia forces in the vector $\mathbf{Q}$.

Equations (1) and (3) constitute a set of $n + m$ mixed DAE's of index three[16], with $\mathbf{q}$ and λ as unknowns. It is a common practice in multibody dynamics to differentiate twice the constraints, thus transforming the equation to index-1, and append the resulting equations to (3) to yield:

$$\begin{bmatrix} \mathbf{M} & \Phi_q^T \\ \Phi_q & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \lambda \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ -\dot{\Phi}_q \dot{\mathbf{q}} - \dot{\Phi}_t \end{Bmatrix} \tag{4}$$

These equations can now be integrated using standard numerical integration techniques with each function evaluation performed using equation (4). In addition, equation (4) may also be easily modified to include Baumgarte stabilization[1].

2.2 *Index-1 Augmented Lagrangian Formulation with Prciections*

Recently, a method has been presented by Bayo and Ledesma[6] which uses an index-1 augmented Lagrangian method with mass-orthogonal projections of the positions and velocities to their constraint manifolds. This formulation leads to the following equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \Phi_q^T \alpha \ddot{\Phi} + \Phi_q^T \lambda^* = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) \tag{5}$$

where λ^* are the Lagrange multipliers. Introducing the expression $\ddot{\Phi} = \Phi_q \ddot{\mathbf{q}} + \dot{\Phi}_q \dot{\mathbf{q}} + \Phi_t$ in equation (5) the following equation is obtained:

$$\left[\mathbf{M} + \Phi_q^T \alpha \Phi_q\right] \ddot{\mathbf{q}} + \Phi_q^T \lambda^* = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) - \Phi_q^T \alpha \left(\dot{\Phi}_q \dot{\mathbf{q}} + \dot{\Phi}_t\right) \tag{6}$$

It is important to note that there is a substantial difference between equation (6) and the Lagrange's multiplier approach represented by equation (4). The leading matrix of equation (4) becomes singular in singular configurations, changing topologies and in the presence of redundant constraints. However, although the mass matrix $\mathbf{M}$ is in general positive semi-definite, the leading matrix of equation (6), $\left[\mathbf{M} + \Phi_q^T \alpha \Phi_q\right]$, is always positive definite, which means that it can always be factored, even in singular positions, changing topologies and/or with redundant constraints[26].

The augmented Lagrangian method comprises a combination of penalty terms and Lagrange multipliers, and they require an iteration procedure for the solution process[27]. The iteration yields the unknown multipliers λ^*, as follows:

$$\lambda_{i+1}^* = \lambda_i^* + \alpha \ddot{\Phi}_{i+1}, \quad i = 0, 1, 2, \ldots \tag{7}$$

with $\lambda_0^* = \mathbf{0}$ for the first iteration. Equation (7) physically represents the introduction at iteration $i + 1$ of forces that tend to compensate for the addition of all the constraint terms which are not exactly zero. Experience shows that when the constraints are scaled to unity, penalty factors ranging from 10^5 to 10^7 give good convergence rates, and only 1 to 2 iterations are required to converge to the solution. The leading matrix remains constant during the iteration process.

As a result of using the index-1 formulation, the solution of equation (6) yields a set of accelerations that not only satisfies dynamic equilibrium but also the constraint conditions $\ddot{\Phi} = \mathbf{0}$. As a consequence, a projection in $\ddot{\mathbf{q}}$ to satisfy the corresponding acceleration constraint conditions is not necessary. On the other hand, the constraints $\Phi = \mathbf{0}$ and $\dot{\Phi} = \mathbf{0}$ may not be satisfied since they have not been directly introduced in the formulation. Consequently, a projection can be done to obtain a new set of $\mathbf{q}$ and $\dot{\mathbf{q}}$ that will satisfy $\Phi = \mathbf{0}$ and $\dot{\Phi} = \mathbf{0}$, respectively. The projections in $\mathbf{q}$ and $\dot{\mathbf{q}}$ may be done as indicated right below.

2.2.1 Projection in q

In order to satisfy the constraint $\Phi = \mathbf{0}$ at each time step, a mass-orthogonal projection of the solution coordinates $\mathbf{q}^*$ (obtained from the numerical integration scheme) on the constraint space is performed. In this

way, the new set of positions $\mathbf{q}$ that are obtained, satisfies the constraints, $\Phi = \mathbf{0}$. This can be achieved by the solution of the following iterative procedure[6]:

$$\left(\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right) \Delta \mathbf{q}^{(i+1)} = -\gamma_p^{(i)} \tag{8}$$

with

$$\gamma_p^{(i)} = \mathbf{M}\left(\mathbf{q}^{(i)} - \mathbf{q}^*\right) + \Phi_{\mathbf{q}}^T \boldsymbol{\lambda}^{(i)} \tag{9}$$

and

$$\mathbf{q}^{(i+1)} = \mathbf{q}^{(i)} + \Delta \mathbf{q}^{(i+1)}, \quad \boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} + \alpha \Phi^{(i+1)} \tag{10}$$

where the superscript indicates the iteration number. Equations (8) through (10) can be used iteratively until $\|\Delta \mathbf{q}\| < \varepsilon$, where ε is a user-specified tolerance.

Note the important feature that the tangent matrix in equation (8) is identical to that used in equation (6) for the dynamic analysis. Furthermore, since the solution $\mathbf{q}$ is close to the initial values $\mathbf{q}^*$, the projection problem can be solved using a modified Newton-Raphson method with no need for updating the tangent matrix (usually convergence is achieved in just one iteration).

2.2.2 Projection in $\dot{\mathbf{q}}$

Similarly, at each time step a mass-orthogonal projection of the velocities $\dot{\mathbf{q}}^*$ on the velocity constraint space is performed to obtain a new set of velocities $\dot{\mathbf{q}}$ that satisfies $\dot{\Phi} = \mathbf{0}$. This can be achieved by the solution of the following equation[6]:

$$\left[\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right] \dot{\mathbf{q}} = \mathbf{M} \dot{\mathbf{q}}^* - \Phi_{\mathbf{q}}^T \left(\alpha \Phi_t + \sigma\right) \tag{11}$$

where the Lagrange multipliers of the projection problem are updated as:

$$\sigma^{(i+1)} = \sigma^{(i)} + \alpha \dot{\Phi}^{(i+1)} \tag{12}$$

A more efficient implementation of the projection of velocities is given by the following recursive set of equations:

$$\left[\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right] \dot{\mathbf{q}}^{(i+1)} = \mathbf{M} \dot{\mathbf{q}}^{(i)} - \Phi_{\mathbf{q}}^T \alpha \Phi_t \tag{13}$$

where the iteration is started by setting,

$$\dot{\mathbf{q}}^{(0)} = \dot{\mathbf{q}}^* \quad \text{and} \quad \sigma^{(0)} = \alpha \dot{\Phi}^{(0)} \tag{14}$$

Since the leading matrix in equation (13) is the same as that of equation (8) used for the projection in $\mathbf{q}$, the velocity projections will require only successive forward-reductions and back-substitutions.

2.3 Index-3 Augmented Lagrangian Formulation with Projections

The index-3 augmented Lagrangian formulation uses mass-orthogonal projections on the velocities and accelerations to their constraint spaces. This method leads to the following equations of motion[6]:

$$\mathbf{M} \ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^T \alpha \Phi + \Phi_{\mathbf{q}}^T \boldsymbol{\lambda}^* = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) \tag{15}$$

where $\boldsymbol{\lambda}^*$ are the Lagrange multipliers. The following iteration process yields the unknown multipliers $\boldsymbol{\lambda}^*$:

$$\boldsymbol{\lambda}_{i+1}^* = \boldsymbol{\lambda}_i^* + \alpha \Phi_{i+1}, \quad i = 0, 1, 2, \ldots \tag{16}$$

with $\boldsymbol{\lambda}_0^* = \mathbf{0}$ for the first iteration. Similar to the index-1 formulation, the value of the penalty factor α affects the convergence rate. Experience shows that, when the constraints are scaled to unity, penalty factors ranging

from 10^7 to 10^8 give good convergence rates for the index-3 formulation, and once again only 1 to 2 iterations are required to converge to the solution.

As a result of using the index-3 formulation, the solution of equation (15) yields a set of $\mathbf{q}_{n+1}$ that not only satisfies dynamic equilibrium but also the constraint conditions $\Phi = 0$. As a consequence, the projection of $\mathbf{q}$ is not necessary, and only $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ need to be projected to satisfy $\dot{\Phi} = 0$ and $\ddot{\Phi} = 0$, respectively. The projection of $\dot{\mathbf{q}}$ may be done as explained above (equations (11) to (14)). The projection of $\ddot{\mathbf{q}}$ is done as follows.

2.3.1 Projections in $\ddot{\mathbf{q}}$

Similar to the velocity analysis, the projection of the generalized accelerations on the constraint manifold can be obtained through the solution of the following equation[6]:

$$\left[\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right]\ddot{\mathbf{q}} = \mathbf{M}\ddot{\mathbf{q}}^* - \Phi_{\mathbf{q}}^T\left\{\alpha\left[\dot{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} + \Phi_t\right] + \kappa\right\} \tag{17}$$

where the associated Lagrange multipliers are updated by the following formula:

$$\kappa^{(i+1)} = \kappa^{(i)} + \alpha\ddot{\Phi}^{(i+1)} \tag{18}$$

Numerical implementation of the augmented Lagrangian formulation for the projection of accelerations is suggested by the recursive set of equations:

$$\left[\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right]\ddot{\mathbf{q}}^{(i+1)} = \mathbf{M}\ddot{\mathbf{q}}^{(i)} - \Phi_{\mathbf{q}}^T \alpha\left\{\dot{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} + \Phi_t\right\} \tag{19}$$

where the recursion is started by setting:

$$\ddot{\mathbf{q}}^{(0)} = \ddot{\mathbf{q}}^* \quad \text{and} \quad \kappa^{(0)} = \alpha\ddot{\Phi}^{(0)} \tag{20}$$

Equation (19) constitutes a system of algebraic equations in $\ddot{\mathbf{q}}$ which is solved iteratively until convergence of the generalized accelerations is achieved. Since the leading matrix in equation (19) is the same as that of equation (13) used for the velocity projection, the acceleration analysis will require only successive forward-reductions and back-substitutions.

3. NUMERICAL SOLUTION OF THE EQUATIONS OF MOTION

During the integration process both formulation schemes, index-1 and index-3, lead to a nonlinear system of simultaneous equations for every time step (equations (6) and (15)) with $\ddot{\mathbf{q}}$ and $\boldsymbol{\lambda}^*$ as primary unknowns. The solution of this set of equations requires an iterative process, commonly based on a Newton-Raphson procedure as it will be shown in the following paragraphs. This procedure is developed by combining (6) and (15) with the difference equations of the numerical integration scheme.

The difference equations of the numerical integration scheme may be expressed as:

$$\dot{\mathbf{q}}_{n+1} = a\mathbf{q}_{n+1} - \hat{\dot{\mathbf{q}}}_{n+1} \quad \text{and} \quad \ddot{\mathbf{q}}_{n+1} = b\mathbf{q}_{n+1} - \hat{\ddot{\mathbf{q}}}_{n+1} \tag{21}$$

where a and b are constants that depend on the numerical integrator and the time step, $\hat{\dot{\mathbf{q}}}_{n+1}$ and $\hat{\ddot{\mathbf{q}}}_{n+1}$ are known quantities that depend on the positions, velocities and accelerations at step n and/or previous steps. Note that if the method is explicit, a and b will be equal to zero.

3.1 Index-1 Augmented Lagrangian Formulation

Substituting equation (21) in the equation of motion (6) for index-1, yields:

$$\left(\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right) b\mathbf{q}_{n+1} + \Phi_{\mathbf{q}}^T \boldsymbol{\lambda}^* + \Phi_{\mathbf{q}}^T \alpha\left(\dot{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} + \Phi_t\right) - \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) - \left(\mathbf{M} + \Phi_{\mathbf{q}}^T \alpha \Phi_{\mathbf{q}}\right)\hat{\ddot{\mathbf{q}}} = \mathbf{0} \tag{22}$$

which constitutes a set of n equations of the form $\mathbf{f}(\mathbf{q}_{n+1}) = 0$ to be solved for the unknown position $\mathbf{q}_{n+1}$. Note that, by introducing the difference equations (21), the positions at step $n+1$ have become the primary

variables as compared with the original index-1 formulation (6) in which the accelerations were the primary unknowns.

A quasi Newton-Raphson procedure can be applied to evaluate the solution of (22) by means of the iterative process:

$$\left(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)^{(i)} \Delta \mathbf{q}^{(i+1)} = -\mathbf{f}\left(\mathbf{q}^{(i)}\right) \tag{23}$$

where the function **f** is defined as:

$$\mathbf{f}\left(\mathbf{q}_{n+1}\right) = \mathbf{M}\ddot{\mathbf{q}}_{n+1} + \Phi_{\mathbf{q}}^{T}\left(\alpha\ddot{\Phi} + \lambda^{*}\right) - \mathbf{Q}_{n+1} \tag{24}$$

and the tangent matrix is approximated by the following matrix:

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}} \equiv b\left(\mathbf{M} + \Phi_{\mathbf{q}}^{T}\alpha\Phi_{\mathbf{q}}\right) - \mathbf{Q}_{\mathbf{q}} - a\mathbf{Q}_{\dot{\mathbf{q}}} \tag{25}$$

where the last two terms of (25) represent the contribution to the tangent matrix coming from the elastic forces (i.e. springs) and those dependent on the velocity terms (i.e. dash-pots). In order to obtain the tangent matrix of equation (25), higher order terms like products of Φ_{qq} with Φ have been neglected. These terms are negligible compared to those of equation (24) (see García de Jalón and Bayo[2], Chapter 8).

3.2 Index-3 Augmented Lagrangian Formulation

The substitution of equation (21) into equation (15) for index-3 yields

$$b\mathbf{M}\mathbf{q}_{n+1} + \Phi_{\mathbf{q}}^{T}\left(\alpha\Phi + \lambda^{*}\right) = \mathbf{Q}_{n+1} + \mathbf{M}\hat{\ddot{\mathbf{q}}}_{n} \tag{26}$$

which again constitutes a set of nonlinear algebraic equations with $\mathbf{q}_{n+1}$ as primary unknowns. A modified Newton Raphson iteration is used again to evaluate the solution by means of the iterative process outlined above with the function **f** defined as:

$$\mathbf{f}\left(\mathbf{q}_{n+1}\right) = \mathbf{M}\ddot{\mathbf{q}}_{n+1} + \Phi_{\mathbf{q}}^{T}\left(\alpha\Phi + \lambda^{*}\right) - \mathbf{Q}_{n+1} \tag{27}$$

The reader should notice that equations (24) and (27) are not the same: the constraints Φ are doubly differentiated in equation (24) whereas they are not in (27). The tangent matrix is approximated by

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}} \equiv b\mathbf{M} + \Phi_{\mathbf{q}}^{T}\alpha\Phi_{\mathbf{q}} - \mathbf{Q}_{\mathbf{q}} - a\mathbf{Q}_{\dot{\mathbf{q}}} \tag{28}$$

3.3 Numerical behavior of the formulations. A Simple Numerical Example.

In order to illustrate the behavior of the methods outlined above for different time steps, we perform the simulation of a double pendulum which moves from at rest conditions in the horizontal position under gravity effects (Figure 1). Each link of the pendulum has a distributed unit mass and a unit length. We use natural coordinates for the modeling process[2] and for the integration we use the trapezoidal rule which is a single step, second order, implicit, *A*-stable method and it is also energy preserving in the linear regime. The total time of simulation is 10 seconds and the penalty factor is 10^{7}.

Table 1 shows the maximum error in the energy norm resulting from each method, index-1 and index-3, for different time steps of integration. It is important to point out that the index-3 formulation does not converge for time steps equal or smaller to 10^{-5} seconds when no pivoting is used. This fact is due, as mentioned above, to the ill-conditioning of the tangent matrix (equation (28)). The problem may be partially circumvented by either increasing the penalty factor or by using a pivoting strategy in the triangularization of the tangent matrix.

Table 1 shows how the pivoting helps the convergence, and extends the range in which the index-3 method can be applied, in this case up to 10^{-5} seconds. However, for time step of 10^{-6} seconds (despite the time taking in the integration) the error starts increasing even with pivoting, and the method lacks robustness.

Conversely, the index-1 method is more robust than the index-3 for time steps size smaller than 10^{-5} seconds. Robustness is meant in the sense that it increases in accuracy, convergence, and does not show numerical ill-conditioning as the time step decreases. On the other hand, since the index-3 formulation is simpler and involves a smaller amount of nonlinear terms is more efficient numerically than the index-1 (see the execution time columns in Table 1). This trend is even more pronounced for large scale systems[25].

It may be concluded, therefore, that a multi-index formulation would take advantage of the good qualities of both methods for different time step sizes. Index-3 behaves well in accuracy and efficiency, and is the right choice for large time steps; however, as the size of the time step is decreased, starting in the neighborhood of $\Delta t=10^{-4}$ seconds, the index-1 method becomes more accurate and robust. A switch mechanism can be implemented within the framework of a variable time stepping algorithm, which will be described next.

Time step size	Index-1		Index-3 Without Pivoting		Index-3 With Pivoting	
	Error	Exec. time	Error	Exec. time	Error	Exec. time
$\Delta t = 10^{-2}$	3.8e-1	0.4 s	2.1e-1	0.4 s	2.2e-1	0.4 s
$\Delta t = 10^{-3}$	4.68e-3	3.14 s	4.6e-3	1.6 s	4.8e-3	1.67 s
$\Delta t = 10^{-4}$	4.3e-5	26.6 s	2.0e-5	15.2 s	4.4e-5	15.9 s
$\Delta t = 10^{-5}$	1.25e-7	254.6 s	Wrong res.	-	3.6e-7	92.5 s
$\Delta t = 10^{-6}$	1.0e-8	2550	Wrong res.	-	1.1e-4	922.9 s

Table 1. Comparative results for the double pendulum using Index-1 and Index-3.

4. A MULTI-INDEX VARIABLE SINGLE STEP METHOD

As mentioned above the index-1 method is more accurate than the index-3 for small time steps, and the index-3 method is more efficient than the index-1 (with same accuracy) for large time steps. Therefore, a multi-index formulation would take advantage of the good qualities of both methods for different time step sizes. Index-3 is the right choice for time steps between 10^{-3} and 10^{-4} (where it is accurate and efficient); however, as we decrease the size of the time step, starting in the neighborhood of $\Delta t = 10^{-4}$ (with no pivoting) or $\Delta t = 10^{-5}$ (with pivoting). the index-1 method becomes more accurate and robust.

When performing a simulation, it is desirable to obtain the most accurate results with the least computational effort. Therefore, it is preferable to use an index-3 approach rather than an index-1 whenever possible (*possible* in this case means when the accuracy in the results is sufficient for the given purposes). Hence, choosing an integration time step is a critical task: if the Δt is too small, the integration process will last more than needed, probably with no evident benefits in the accuracy of the numerical results. Conversely, if the time step is too large, the integration process may diverge or the results obtained may be wrong. Consequently, the optimum combination of integration time step and index approach for each problem should be sought.

Considering a single and fixed time step algorithm, the best combination of index approach and time step will be determined by the nature of the problem and by the worst function conditioning in the sense of its time history. That is, if the function being integrated is smooth all over the time interval, a moderate time step size combined with an index-3 approach will be a good choice. However, in cases where the function has sharp zones, even of short duration, the whole integration process would have to be carried out using a small time step, otherwise it would fail. The choice for the right index would be determined by the step size. As a general rule if the time step size is smaller than 10^{-5} seconds an index-1 should be the choice. Conversely, for time step sizes over 10^{-4} seconds, the choice should be index-3. There is a gap between 10^{-4} and 10^{-5} in which both methods perform well and the choice becomes irrelevant.

A variable time step integrator would certainly lead to an economy in CPU time for the case considered above. If the time step is variable, then it will be possible to accommodate it to obtain constant local error or to maintain this error below some given reference value. Classical approaches in ODE establish a variable time step strategy based on a measure of the local truncation error, which is determined by evaluating either the integrated functions using two different order methods (the Runge-Kutta-Fehlberg, for example), or by integrating two successive time intervals with step size h_n and $h_n/2$. Any of these strategies require many more extra function evaluations, thus compromising the numerical efficiency of the method.

A first thought is to use the kinetic energy stored in the penalty system as a measure of the local integration error. This energy is evaluated in the following way:

$$K_\alpha = \frac{1}{2}\alpha\dot{\Phi}^T\dot{\Phi} \tag{29}$$

which can be computed easily and efficiently. Figures 2, 3 and 4 show the vertical acceleration of the end point in the double bar pendulum, the penalty system kinetic energy, and the total system energy, respectively. It may be observed that the acceleration is very spiky because the second bar of the double pendulum undergoes a very strong motion. Also, when sudden changes in the acceleration take place the penalty kinetic energy increases and the total system energy varies. Hence, qualitatively the energy in the penalty system can be taken as a measure of the local integration error, provided it increases when the system encounters more difficulties in satisfying the constraints.

Although good in principle, this manner of evaluating the local integration error has an important drawback: it is quite difficult to establish a general quantitative relationship between the local integration error and the energy stored in the penalty system.

Due to this reason, and taking into account the existing correlation between the energy and the local error (see Figure 4 and Figure 5), we propose to use an integral of motion (system total energy in conservative systems) as a measure of the local integration error. For non-conservative systems and considering the use of fully Cartesian coordinates, the following integral of motion (energy invariant) may be established: premultiplying the equations of motion by $\dot{\mathbf{q}}^T$ and integrating over time the following expression may be obtained:

$$\int_0^t \dot{\mathbf{q}}^T\left(\mathbf{M}\ddot{\mathbf{q}} + \boldsymbol{\Phi}_\mathbf{q}^T\boldsymbol{\lambda} - \mathbf{Q}\right)d\tau = constant \tag{30}$$

Since the constraint forces are composed of sets of internal forces that do not produce work, the integration of this equation leads to:

$$\Pi = T(t) + V(t) - \int_0^t \dot{\mathbf{q}}^T\mathbf{Q}d\tau = constant \tag{31}$$

Equation (31) clearly yields $T + V = constant$ for conservative systems.

5. THE MULTI-INDEX AUTO-TIME-STEPPING ALGORITHM

Using the energy invariant described in the previous section (equation (31)) as a measure of the local integration error, a variable time step size strategy is proposed. The key idea is to modify the step size when the change in the energy invariant exceeds an allowable value. When this occurs a new time step size is calculated in order to achieve the desired accuracy. The new time step size $\overline{\Delta t}_n$ can be expressed as a function of the previous time step size Δt_n *in* the following way[28]:

$$\overline{\Delta t}_n = \eta_r \Delta t_n \tag{32}$$

where η_r, is a parameter defined as:

$$\eta_r = \left(\frac{\nu\varepsilon h_n}{|T_r|}\right)^{1/r} \tag{33}$$

where T_r is a function of the local error estimation; ε is the allowed local error; r the order of the integrator (it should be set to 2 for the trapezoidal rule) and ν a safety factor typically taken as 0.8. In fact, good results are obtained for values of the parameter r in the range $0.25 \le 1/r \le 0.5$.

The strategy to perform the change of index is based on the general consideration that when integrating in index-3, the larger the time step size, the better the performance, and that the time step should never be smaller than 10^{-4}. On the contrary, for the index-1 approach, the smaller the step size, the better the results. There is an overlapping zone, ranging from $\Delta t = 10^{-4}$ to 10^{-5} in which both approaches are suitable. In this zone, it is preferable to use the more economical index-3 rather than the index-1 approach.

Figure 6 illustrates in a flowchart manner the general strategy established for the modification of the time step size. The index change is also based on convergence considerations. If any integration step does not converge, or the local error is larger than a specified allowable value U_r, a time step reduction is performed. If the

problem persists, using the index-3, after several consecutive time step reductions (denoted as *maxite* in the flowchart of Figure 6), even for step sizes larger than 10^{-4} then it is necessary to change to index-1. On the contrary, when the local error is below some specified value L_r the step size is enlarged and a change of index is performed (from index-1 to index-3) provided the step size is larger than 10^{-4}. Based on empirical evidence the estimated values for U_r and L_r are around 10^{-5} and 10^{-7}, respectively.

It is important to remark that prior to the change to index-1 from index-3, it is necessary to recalculate the accelerations $\ddot{\mathbf{q}}$ and velocities $\dot{\mathbf{q}}$ to satisfy equation (5) for the current positions $\mathbf{q}$.

6. NUMERICAL RESULTS

In this section two examples are presented. First of all the double pendulum described earlier in this paper and the front suspension system of an off-road vehicle.

6.1 Double Pendulum

As it has been written before, the *double pendulum case* consists on the simulation of the behavior of a double pendulum released from its horizontal resting position for a total time period of 10 seconds. Each bar of the pendulum has a length of 1 meter and a uniform mass of 1 Kg.

Figure 7 and Figure 8 show the results obtained in this simulation. It may be seen that index and time step size changes occur when sudden variations in the system energy invariant take place. In this simple example the overhead introduced by the variable-time-stepping algorithm is comparatively large to the total CPU time consumed. Hence no benefits, in terms of CPU time, are obtained. However, considerable accuracy is obtained when integrating at variable time step size, as seen in Figure 8, when compared to fixed time step size.

6.2 Off-road vehicle suspension system

The 1/4 ton 4x4 Iltis vehicle[29] has been proposed as a benchmark problem by the European automobile industry to check multibody dynamic codes.

We perform the simulations using the front suspension of the vehicle (see Figure 9). The model requires a total of 23 variables, related through 22 constraint equations, since there is only one degree of freedom. The characteristics of the suspension are:

- a *leaf spring*, modeled as a linear spring of stiffness 35,900 N/m for deformations smaller than 14.5 centimeters. When such deformation is exceeded the suspension hits a second spring of value 10^7 N/m, thus adding a much higher stiffness to the suspension;
- a *shock absorber*, which provides an elastic force in Newtons due to an external polymer, given by,

$$F_S = -4.0092 \cdot 10^6 + 2.8397 \cdot 10^7 x - 6.7061 \cdot 10^7 x^2 + 5.2796 \cdot 10^7 x^3 \tag{34}$$

and a damping force also in Newtons given by the following formula:

$$\begin{aligned} F_D &= -208.21 + 922.15v \quad v < -0.2 \text{ m/s} \\ F_D &= 4972.8135v + 16977.86v^2 - 29916.125v^3 - 197825.5v^4 \quad -0.2 < v < 0.21 \text{ m/s} \\ F_D &= 959.5819 + 817.3635v \quad v > 0.21 \text{ m/s} \end{aligned} \tag{35}$$

where the distance x is in meters;

- a *tire*, modeled by means of a linear vertical spring of stiffness 460,000 N/m.

The simulation is performed as follows: the suspension starts moving from at rest conditions at a speed of 5 m/s, and its position does not correspond to the static equilibrium. Thus, it freely oscillates until the static equilibrium position is reached. After three seconds, the suspension goes over a road bump, defined by a cosine function, and afterwards it freely oscillates until the equilibrium is reached again. The complete analysis lasts for 6 seconds.

This simulation is carried out under two different conditions. The first one is done using a index-3 approach with a fixed time step of 10^{-2} seconds. The second simulation is performed at variable time step and multi-index, starting with index-3 and a step size of 10^{-3} seconds. The tolerance values U_r and L_r are set to 10^{-6} and 10^{-8}, respectively. The first simulation took 8.41 seconds of CPU time and the second took 36.07 seconds.

Figure 10 shows the time history of the time step size. It may be clearly seen that, for the specified tolerance it is necessary to shorten the time step size only when the wheel is passing over the bump. At the beginning the step size is quickly adjusted to speed up the process while maintaining the local error between the allowable limits.

Figure 11 and Figure 12 show the total system energy in both cases. It is important to remark that only when translating the origin and scaling the energy represented in the vertical axis (Figure 12) one can notice its change. The biggest deviation from the initial value (425,128 J vs. 100 J) is about 0.02% of the total system energy in both cases.

It is important to remark that similar precision in total system energy conservation can be obtained performing the simulation with an index-1 approach and a fixed time step size of $5 \cdot 10^{-4}$ seconds. In that case the simulation takes 155 seconds (over 400% more than in the variable index-variable step case), thus corroborating the benefits of using the proposed multi-index variable time step procedure.

As a result of this example, it is clear that a multi-index-1-3 approach with variable time step provides an excellent solution, since the more efficient index-3 method is used during a large part of the motion, and the index-1 only at that part where the time step decreases below the critical limit where the index-3 loses precision and accuracy.

7. CONCLUSIONS

The proposed multi-index formulation shows a behavior that could be defined as complementary. On one hand, index-3 is very accurate and more efficient than index-1 for large time steps, whereas, index-3 provides wrong results for very small time steps due to numerical ill conditioning. In order to accommodate all these features a multi-index variable time step size strategy has been devised based on the following criteria:

1. Whenever possible the time step is increased.
2. The time step size is decreased based on the measure of the total energy invariant, which has been showed to maintain a relationship with the local error.
3. The threshold to change from index-3 to index-1 is set to 10^{-4}.

These are the major conclusions:

- The system energy invariant has proved to be a good measure of the local integration error. This feature makes unnecessary the use of traditional (but less efficient) error criteria used in standard numerical integration methods.
- The variable time step strategy and the switch from one method to the other do not cause problems during the integration process.
- The use of the proposed technique allows a substantial speedup gain in CPU time for large scale systems, thus helping to achieve real-time behavior.
- The method is general and can also be applied to solve the dynamics of flexible multibodies.

ACKNOWLEDGMENTS

The support of this work provided by the Army Research Office under grant DAAH04-95-1-0662, and the CICYT of the Spanish Ministry of Education under grant TAP95-0226 is gratefully acknowledged.

LIST OF FIGURES

Figure 10. Time history of the time step size.

Figure 11. Total system energy.

Figure 12. Amplification of the total system energy.

REFERENCES

1. E.J. Haug, Computer-Aided Kinematics and Dynamics of Mechanical Systems, Volume I: Basic Methods, Allyn and Bacon. Newton, MA, 1989.
2. J. García de Jalón and E. Bayo, *Kinematic and Dynamic Simulation of Multibody Systems. The Real-Time Challenge*, Springer-Verlag. New York, 1994.
3. P.E. Nikravesh, *Computer-Aided Analysis of Mechanical Systems*, Prentice-Hall. Englewood Cliffs, N.J., 1988.
4. A.A. Shabana, *Dynamics of Multibody Systems*, Wiley. New York, 1989.
5. J. Baumgarte, 'Stabilization of Constraints and Integrals of Motion in Dynamical Systems', *J. of Comp. Meth. in App. Mech. and Eng.*, **1**, 1–16 (1972).
6. E. Bayo and R. Ledesma, 'Augmented Lagrangian and Mass-Orthogonal Projection Methods for Constrained Multibody Dynamics', *J. of Nonlinear Dynamics*, **9**, 113–130 (1996).
7. A. J. Kurdila, J. L. Junkins and S. Hsu, 'Lyapunov Stable Penalty Methods for Imposing Nonholonomic Constraints in Multibody System Dynamics', *Nonlinear Dynamics*, **4**, 51–82 (1993).
8. R. A. Wehage and E. J. Haug, 'Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems', *ASME Journal of Mechanical Design*, **104**, 247–255 (1982).
9. T. R. Kane and D. A. Levinson, *Dynamics: Theory and Applications*, McGraw-Hill. New York, 1985.
10. M. A. Serna, R. Avilés and J. García de Jalón, 'Dynamic Analysis of Planar Mechanisms with Lower-Pairs in Basic Coordinates', *Mechanism and Machine Theory*, **17**, 397–403 (1982).
11. W. Jerkovsky, 'The Structure of Multibody Dynamic Equations', *J. of Guidance and Control*, **1**, 173–182 (1978).
12. S. S. Kim and M. J. Vanderploeg, 'A General and Efficient Method for Dynamic Analysis of Mechanical Systems Using Velocity Transformations', *J. of Mechanisms, Transmissions and Automation in Design*, **108**, 176–182 (1986).
13. P. E. Nikravesh and G. Gim, 'Systematic Construction of the Equations of Motion for Multibody Systems Containing Closed Kinematic Loops', *Advances in Design Automation*, **3**, 27–33, (1989).
14. D. S. Bae and Y. S. Won, 'A Hamiltonian Equation of Motion for Real-Time Vehicle Simulation', *Advances in Design Automation*, **2**, 151–157 (1990).
15. A. Avello, J. M. Jiménez, E. Bayo and J. García de Jalón, 'A Simple and Parallelible Method for Real-Time Dynamic Simulation Based on Velocity Transformations', *J. of Comp. Meth. in App. Mech. and Eng.*, **107**, 313–339 (1993).
16. K. E. Brenan, S. L. Campbell and L. R. Petzold, *The Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, Elsevier Science Publishing. Co., North-Holland. New York, 1989.
17. E. Griepentrog, M. Hanke and R. Marz, *Berlin Seminar on Differential Algebraic Equations*, Fachbereich Mathematik der Humboldt-Universitat zu Berlin. Berlin, 1992.
18. C. Führer, and B. J. Leimkuhler, 'Numerical solution of Differential Algebraic Equations for Constrained Mechanical Motion', *Numerische Mathematik*, **59**, 55–69 (1991).
19. E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems*, Springer-Verlag. Berlin, 1996.
20. Ch. Lubich, 'h2-extrapolation methods for differential-algebraic systems', *Impact Comput. Sc. Eng.*, **1**, 260–268 (1989).
21. Ch. Lubich, U. Nowak, U. Pöhle and Ch. Engstler, *MEXX–Numerical Software for the Integration of Constrained Mechanical Multibody Systems*, preprint SC 92-12, Konrad-Zuse-Zentrum. Berlin, 1992.
22. L. R. Petzold, 'A Description of DASSL: A Differential-Algebraic System Solver', *Proceedings of IMACS World Congress*. Montreal, Canada (1982).
23. M. Arnold, 'Stability of Numerical Methods for Differential-Algebraic Equations of Higher Index', *App. Numerical Mathematics*, **13**, 5–14 (1993).
24. M. Arnold, 'Half-explicit Runge-Kutta methods with explicit stages for differential-algebraic systems of index 2', *to be published*. (1996)
25. J. Cuadrado, J. Cardenal, and E. Bayo, 'Modeling and solution methods for efficient real-time simulation of multibody dynamics', *Multibody System Dynamics*, **1**, 259–280 (1997).
26. E. Bayo and A. Avello, 'Singularity Free Augmented Lagrangian Algorithms for Constraint Multibody Dynamics', *Nonlinear Dynamics*, **5**, 209–231 (1994).
27. D. P. Bertsekas, *Constraint Optimization and Lagrange Multipliers Methods*, Academic Press. New York, 1982.
28. A. Ralston and P. Rabinowitz, *A First Course in Numerical Analysis. 2nd Edition*, McGraw-Hill, Inc. New York. 1978.
29. 'Iltis Data Package', *IAVSD Workshop*, Herbertov. Czechoslovakia (1990).

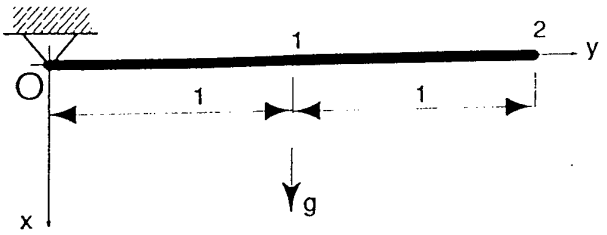


Figure 1. Double pendulum initial position.

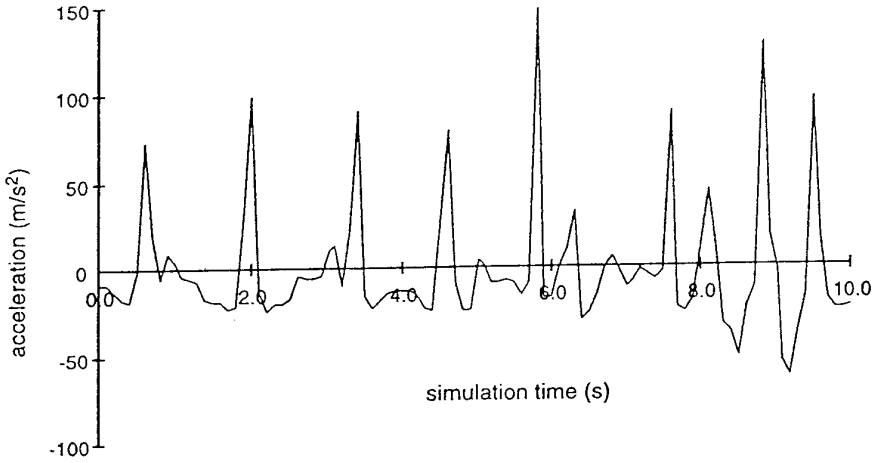


Figure 2. Vertical acceleration of the double pendulum end point.

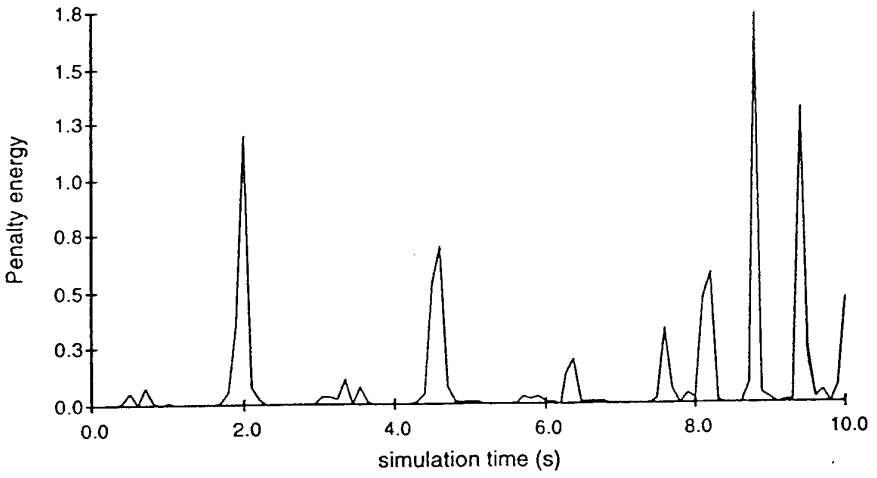


Figure 3. Double pendulum kinetic energy of penalty system.

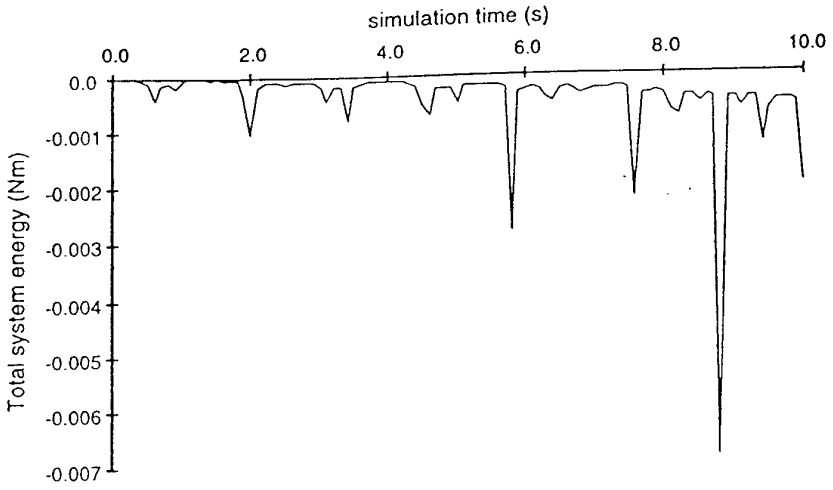


Figure 4. Total system energy vs. time

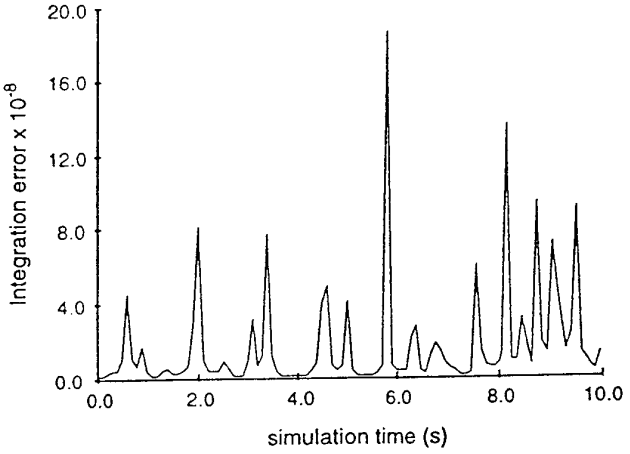
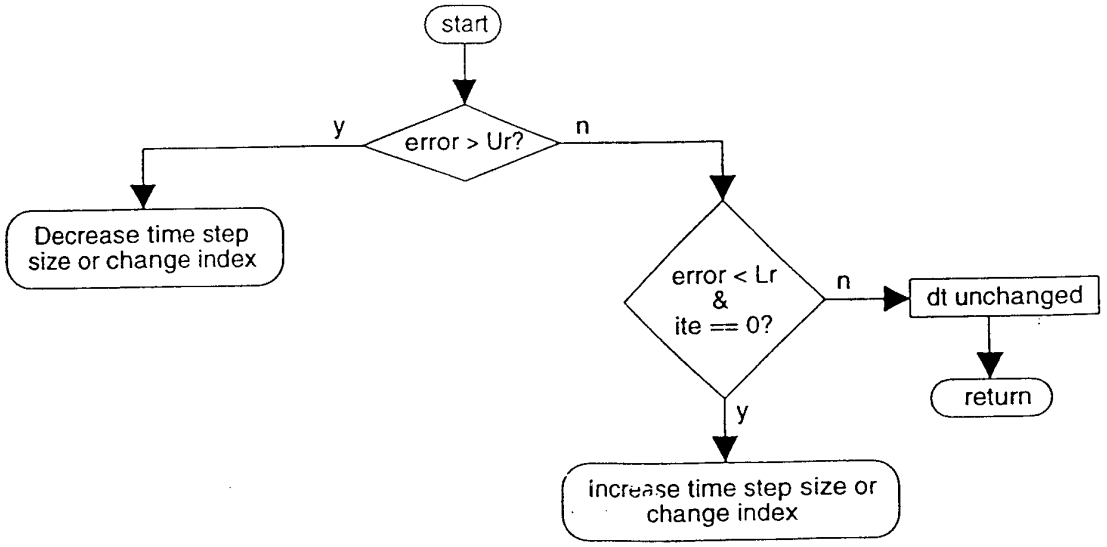
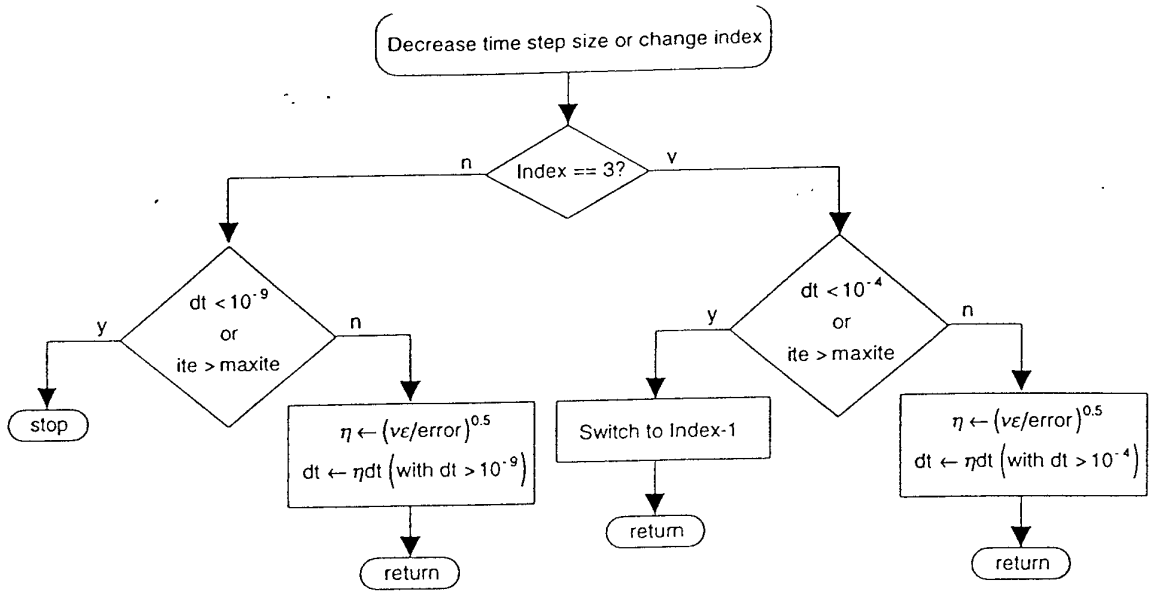


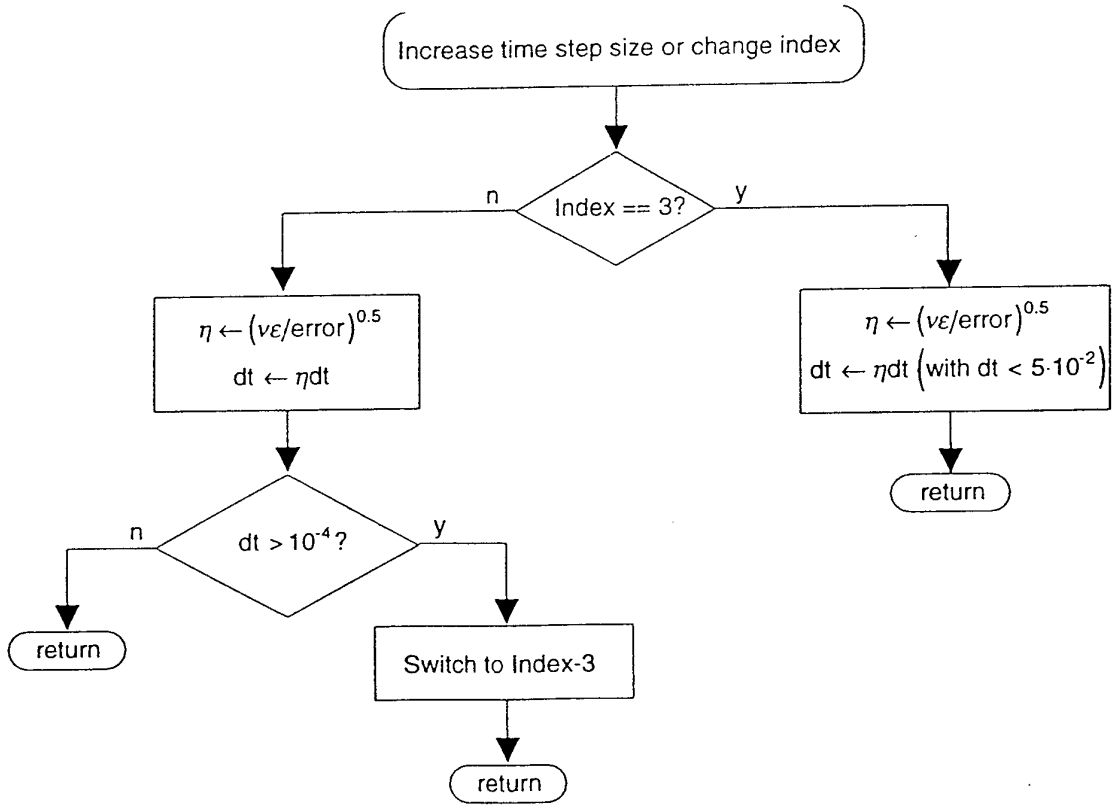
Figure 5. Local integration error vs. time



6.a)



6.b)



6.c)

Figure 6. Multi-index and time-stepping strategy flowchart block diagram.

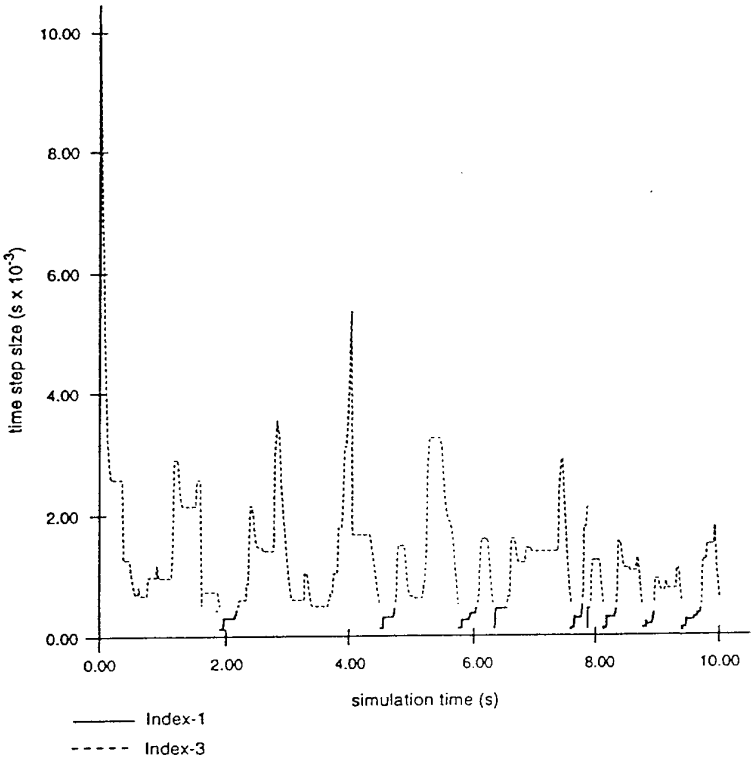


Figure 7. Time history of the time step size.

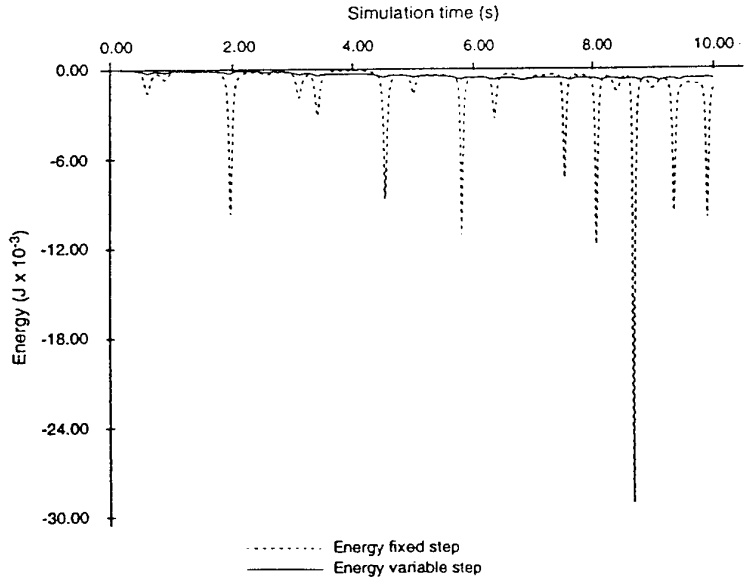


Figure 8. Total system energy.

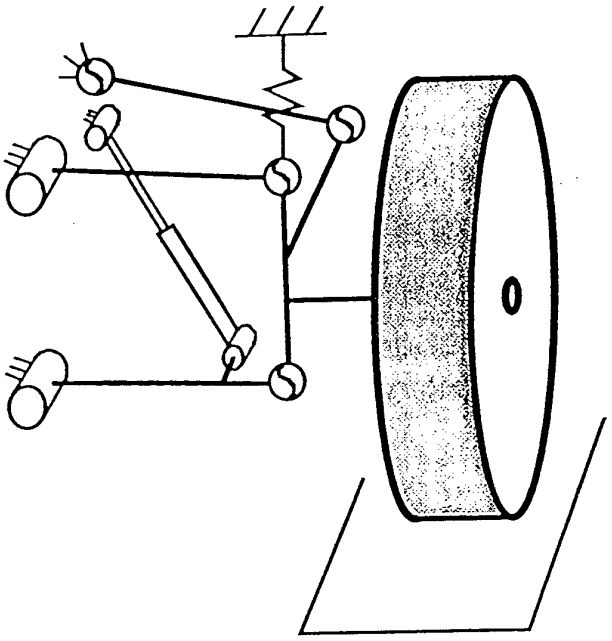


Figure 9. Model of the Iltis suspension system.

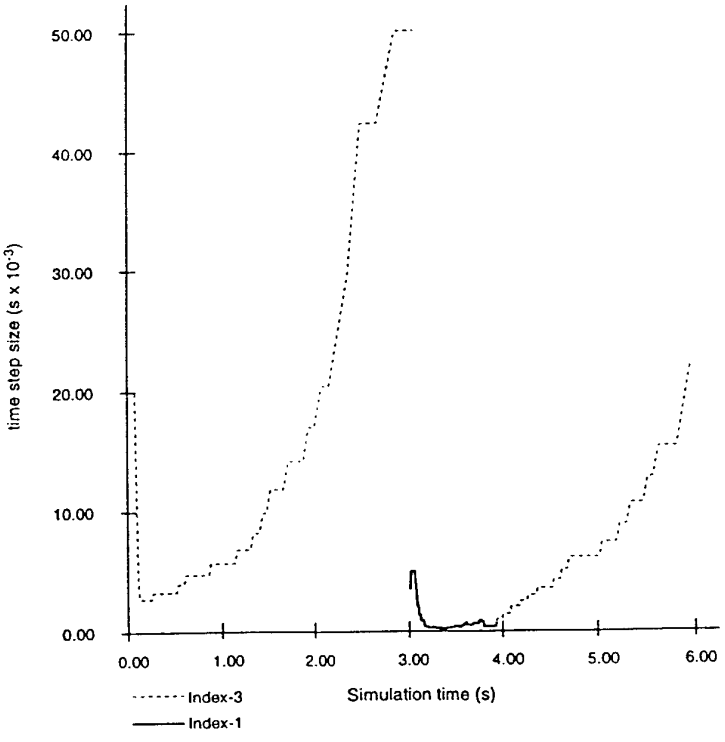


Figure 10. Time history of the time step size.

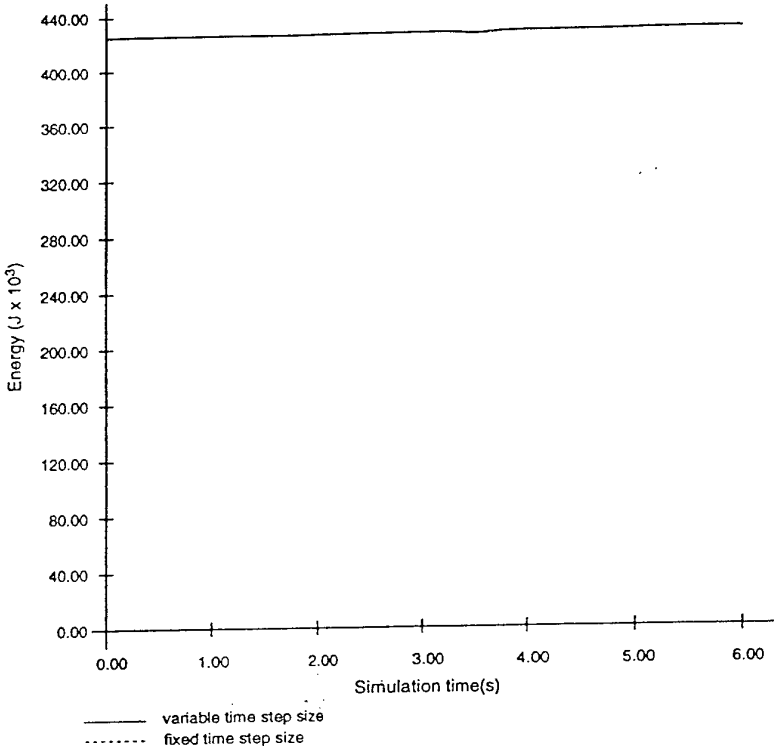


Figure 11. Total system energy.

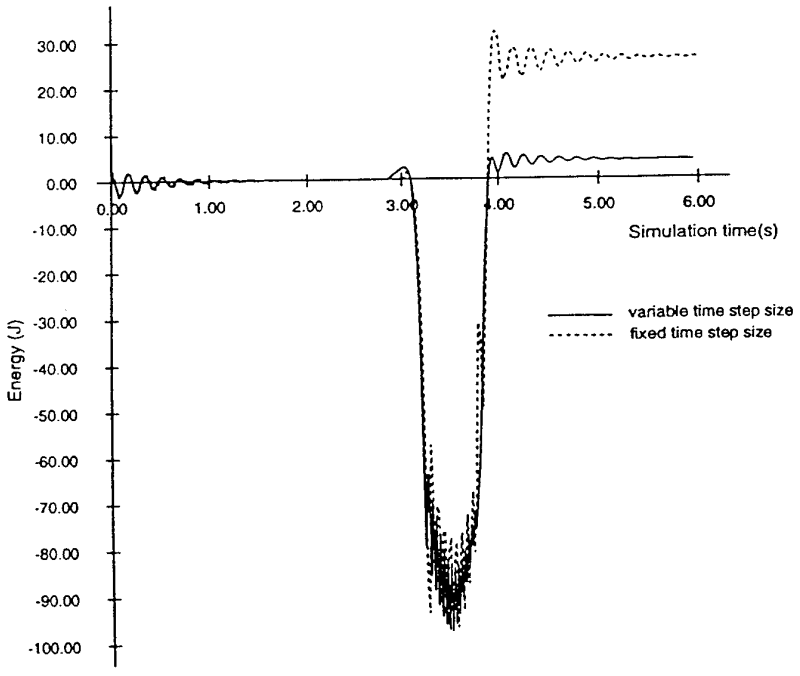


Figure 12. Amplification of the total system energy.

PUBLICATION 3

Intelligent Simulation of Multibody Dynamics: Space-State and Descriptor Methods in Sequential and Parallel Computing Environments

J. CUADRADO, J. CARDENAL and P. MORER
Escuela Politécnica Superior. University of La Coruña. Mendizábal, s/n. 15403 Ferrol. Spain.
E. BAYO
Escuela Politécnica Superior. University of La Coruña. Mendizábal, s/n. 15403 Ferrol. Spain; and School of Engineering and Architecture. University of Navarre. 31080 Pamplona. Spain.

Abstract. Real-time dynamic simulation of large, realistic and complex multibody systems is essential to develop modern technologies such as virtual prototyping, man-in-the-loop simulators and intelligent vehicle control systems. In order to achieve real-time performance, current commercial codes require the use of large costly computers, thus limiting the number of potential users.

This paper shows that real-time can be achieved on medium-size workstations if, on one hand, an adequate combination of modeling, dynamic formulation and numerical integration scheme is selected and, on the other hand, advantage is taken from sparse matrix technology and parallel computing. A study of space-state and descriptor methods involving the dynamics of a whole vehicle model is carried out, and, as conclusion, two methods are proposed as the best candidates for real-time simulation.

Key words: real-time-simulation, multibody dynamics, solution algorithms, parallel computing, sparse matrix technologies.

1. Introduction and Background

Simulation tools are becoming more and more relevant in mechanical design as they allow for a reduction of the design-cycle, thus leading to products at reduced costs and with earlier presence in the market place. More specifically, computer aided dynamics of multibody systems is of great interest for automotive, aerospace, robotics, biomechanical and military industries. In a number of applications, the time spent by the computer in performing the analysis is not a crucial matter. However, there are applications that cannot be developed unless real-time simulation is achieved. This group encompasses hardware-in-the-loop settings, where a real component may be tested; man-in-the-loop devices, used for training purposes; virtual prototyping, that allows the designer to interact with his model and immediately see the results of a what-if analysis; and intelligent vehicle control systems, that lead to safer and more comfortable transportation vehicles. Available codes and algorithms are capable of reaching real-time performance on conventional computers when managing small and academic examples, but large, realistic and complex models lead to very large calculation efforts which require powerful computers and efficient codes to meet real-time.

In a previous paper [1], the authors carried out a study on the different factors involved in the simulation process: modeling, dynamic formulation and integration procedures; and concluded that the adequate combination of these factors depends on the properties of the system to be analyzed. Therefore, it is not correct to talk about the best method for all cases but about the best method for a

particular problem, and consequently the concept of intelligent simulation was introduced. In [1], four methods were developed and tested to find out what kind of problems they managed better. For this purpose, a benchmark library containing open and closed kinematic loops, mechanisms with singular configurations and stiff systems was also set up. Results led to conclude that two of the methods were notably superior to the others: an index-3 augmented Lagrangian formulation with mass-orthogonal projections (a descriptor method), and a fully-recursive formulation based on relative coordinates (a state-space method). However, none of the examples examined in [1] was large enough to be able to generalize the conclusions for complex systems. Realistic models of vehicles, satellites, robots or humans usually exceed 150 coordinates in size. Furthermore, parallel computing and sparse matrix solvers, whose influence is noticed when the problem size increases, are factors to be taken into account which had not been considered.

These are the reasons that led to develop the work presented in this paper. The two methods mentioned above, and a third one, an index-3 classical Lagrangian formulation, commonly used in commercial codes, are described in detail. Emphasis is made on the implementation aspects which are crucial, and constitute a major improvement with respect to reference [1]. Then, the dynamics of the complete model of a 4x4 military vehicle composed of 168 coordinates undergoing several demanding maneuvers, are solved using the three methods under comparison. The influence of sparse matrix technology and parallel computing is also examined. Based on the results, it is concluded that real-time performance can be reached on medium-size workstations.

2. Dynamic Formulations and Numerical Implementation

In this section, the three methods whose performances are to be compared will be briefly described. Description of each method includes the form of the dynamic equations and the integration scheme.

2.1. INDEX-3 AUGMENTED LAGRANGIAN FORMULATION WITH PROJECTIONS

The index-3 augmented Lagrangian formulation with mass-orthogonal projections may be found in [2]. The corresponding equations of motion are given by

$$\mathbf{M}\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\Phi + \Phi_{\mathbf{q}}^{\mathrm{T}}\lambda^{*} = \mathbf{Q} \qquad (1)$$

where $\mathbf{M}$ is the mass matrix, $\ddot{\mathbf{q}}$ are the accelerations, $\Phi_{\mathbf{q}}$ the Jacobian matrix of the constraint equations, α the penalty factor, $\mathbf{\Phi}$ the constraints vector, λ^{*} the Lagrange multipliers and $\mathbf{Q}$ the vector of applied and velocity dependent inertia forces. The Lagrange multipliers are obtained from the following iteration process [2],

$$\lambda_{i+1}^{*} = \lambda_{i}^{*} + \alpha\Phi_{i+1}, \quad i=0,1,2,\ldots \qquad (2)$$

In [2], the value $\lambda_{0}^{*} = 0$ was chosen to start the iteration. However, more recent experiences have demonstrated that better convergence is attained when extrapolating λ_{0}^{*} from the Lagrange multipliers

already calculated in previous time steps (note that sub-index n indicates the time step, and sub-index i refers to the iteration step within a time step)

$$\left(\lambda_o^*\right)_{n+1} = 2\lambda_n - \lambda_{n-1} \tag{3}$$

As integration scheme, the implicit single-step trapezoidal rule has been adopted. The corresponding difference equations in velocities and accelerations are:

$$\dot{\mathbf{q}}_{n+1} = \frac{2}{\Delta t}\mathbf{q}_{n+1} + \hat{\dot{\mathbf{q}}}_n \quad \text{with} \quad \hat{\dot{\mathbf{q}}}_n = -\left(\frac{2}{\Delta t}\mathbf{q}_n + \dot{\mathbf{q}}_n\right) \tag{4}$$

$$\ddot{\mathbf{q}}_{n+1} = \frac{4}{\Delta t^2}\mathbf{q}_{n+1} + \hat{\ddot{\mathbf{q}}}_n \quad \text{with} \quad \hat{\ddot{\mathbf{q}}}_n = -\left(\frac{4}{\Delta t^2}\mathbf{q}_n + \frac{4}{\Delta t}\dot{\mathbf{q}}_n + \ddot{\mathbf{q}}_n\right) \tag{5}$$

Dynamic equilibrium can be established at time step $n+1$ by introducing the difference equations (4) and (5) into the equations of motion (1), thus yielding

$$\frac{4}{\Delta t^2}\mathbf{M}\mathbf{q}_{n+1} + \Phi_{\mathbf{q}\,n+1}^{\mathrm{T}}\left(\alpha\Phi_{n+1} + \lambda_{n+1}\right) - \mathbf{Q}_{n+1} + \mathbf{M}\hat{\ddot{\mathbf{q}}}_n = 0 \tag{6}$$

For numerical reasons, the scaling of equation (6) by a factor of $\frac{\Delta t^2}{4}$ seems to be convenient, thus yielding

$$\mathbf{M}\mathbf{q}_{n+1} + \frac{\Delta t^2}{4}\Phi_{\mathbf{q}\,n+1}^{\mathrm{T}}\left(\alpha\Phi_{n+1} + \lambda_{n+1}\right) - \frac{\Delta t^2}{4}\mathbf{Q}_{n+1} + \frac{\Delta t^2}{4}\mathbf{M}\hat{\ddot{\mathbf{q}}}_n = 0 \tag{7}$$

or, symbolically $\mathbf{f}(\mathbf{q}_{n+1}) = 0$.

In order to obtain the solution of this non-linear system, the widely used iterative Newton-Raphson method may be applied

$$\left[\frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}\right]_i \Delta\mathbf{q}_{i+1} = -\left[\mathbf{f}(\mathbf{q})\right]_i \tag{8}$$

where

$$\left[\mathbf{f}(\mathbf{q})\right] = \frac{\Delta t^2}{4}\left(\mathbf{M}\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\Phi + \Phi_{\mathbf{q}}^{\mathrm{T}}\lambda^* - \mathbf{Q}\right) \tag{9}$$

and the approximated tangent matrix is:

$$\left[\frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}\right] = \mathbf{M} + \frac{\Delta t}{2}\mathbf{C} + \frac{\Delta t^2}{4}\left(\Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\Phi_{\mathbf{q}} + \mathbf{K}\right) \tag{10}$$

where **C** and **K** represent the contribution of damping and elastic forces of the system provided they exist.

A closer look at the tangent matrix reveals that ill-conditioning may appear when the time step becomes small. It may be seen in equation (10) that **K** and the constraint terms are multiplied by Δt^2, **C** by Δt and **M** is not affected by the step size. As a consequence when Δt reaches small values, large round off errors will occur. In fact, it has been demonstrated in [3] that for an index-3 differential algebraic equation the tangent matrix has a condition number of order $1/\Delta t^3$. Consequently, the method is bound to have round-off errors for step sizes smaller than 10^{-5}.

The procedure explained above yields a set of positions $\mathbf{q}_{n+1}$ that not only satisfies the equations of motion (6), but also the constraint conditions $\Phi = 0$. However, it is not expected that the corresponding sets of velocities and accelerations satisfy $\dot{\Phi} = 0$ and $\ddot{\Phi} = 0$, because these conditions have not been imposed in the solution process. To overcome this difficulty, mass-orthogonal projections in velocities and accelerations have been proposed in [2]. However, we propose a modification to the method of [2] in order to get as leading matrix the same tangent matrix appearing in equation (10). As a consequence, triangularization is avoided and projections in velocities and accelerations are performed with just a forward reduction and a back substitution.

If $\dot{\mathbf{q}}^*$ and $\ddot{\mathbf{q}}^*$ are the velocities and accelerations obtained after convergence has been achieved in the Newton-Raphson iteration, their cleaned counterparts $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are calculated from

$$\left[\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\left(\Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\Phi_{\mathbf{q}}+\mathbf{K}\right)\right]\dot{\mathbf{q}}=\left[\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\mathbf{K}\right]\dot{\mathbf{q}}^*-\frac{\Delta t^2}{4}\Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\Phi_t \qquad (11)$$

for the velocities, and

$$\left[\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\left(\Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\Phi_{\mathbf{q}}+\mathbf{K}\right)\right]\ddot{\mathbf{q}}=\left[\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\mathbf{K}\right]\ddot{\mathbf{q}}^*-\frac{\Delta t^2}{4}\Phi_{\mathbf{q}}^{\mathrm{T}}\alpha\left(\dot{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}+\dot{\Phi}_t\right) \qquad (12)$$

for the accelerations.

Sparse matrix technology has been implemented to solve all the linear sets of equations that arise when applying this method. The fact that the leading matrix is symmetric and positive-definite has been taken into account as well.

2.2. INDEX-3 CLASSICAL LAGRANGIAN FORMULATION WITH PROJECTIONS

An equivalent statement may be developed for the classical Lagrangian formulation [4]. In this case, the equations of motion are

$$\mathbf{M}\ddot{\mathbf{q}}+\Phi_{\mathbf{q}}^{\mathrm{T}}\lambda=\mathbf{Q} \qquad (13)$$

$$\Phi=0 \qquad (14)$$

where the former relation express the dynamic equilibrium and the latter one contains the constraints among variables.

Again the trapezoidal rule is chosen to solve the initial value problem. The combination of the difference equations (4) and (5) with equations (13) and (14) leads to:

$$\frac{4}{\Delta t^2}\mathbf{M}\mathbf{q}_{n+1} + \Phi_{\mathbf{q}_{n+1}}^{\mathrm{T}}\lambda_{n+1} - \mathbf{Q}_{n+1} + \mathbf{M}\hat{\ddot{\mathbf{q}}}_n = 0 \tag{15}$$

$$\Phi_{n+1} = 0 \tag{16}$$

Equations (15) and (16) are set at time step $n+1$. As for the augmented Lagrangian formulation described before, these equations are scaled by a factor of $\frac{\Delta t^2}{4}$, yielding

$$\mathbf{M}\mathbf{q}_{n+1} + \frac{\Delta t^2}{4}\Phi_{\mathbf{q}_{n+1}}^{\mathrm{T}}\lambda_{n+1} - \frac{\Delta t^2}{4}\mathbf{Q}_{n+1} + \frac{\Delta t^2}{4}\mathbf{M}\hat{\ddot{\mathbf{q}}}_n = 0 \tag{17}$$

$$\frac{\Delta t^2}{4}\Phi_{n+1} = 0 \tag{18}$$

Equations (17) and (18) may be written symbolically as $\mathbf{f}(\mathbf{q}_{n+1}) = 0$ and then solved by means of a Newton-Raphson procedure with

$$[\mathbf{f}(\mathbf{q})] = \begin{Bmatrix} \frac{\Delta t^2}{4}\left(\mathbf{M}\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{\mathrm{T}}\lambda - \mathbf{Q}\right) \\ \frac{\Delta t^2}{4}\Phi \end{Bmatrix} \tag{19}$$

and the tangent matrix approximated by

$$\left[\frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}\right] = \begin{bmatrix} \mathbf{M} + \frac{\Delta t}{2}\mathbf{C} + \frac{\Delta t^2}{4}\mathbf{K} & \frac{\Delta t^2}{4}\Phi_{\mathbf{q}}^{\mathrm{T}} \\ \frac{\Delta t^2}{4}\Phi_{\mathbf{q}} & 0 \end{bmatrix} \tag{20}$$

Since only constraint conditions in positions have been imposed, it is not expected that first and second derivatives of the constraints are satisfied by velocities and accelerations. Similar to the augmented Lagrangian formulation seen above, mass-orthogonal projections may be performed in such a way that the corresponding leading matrices are identical to the tangent matrix (20). If again $\dot{\mathbf{q}}^*$ and $\ddot{\mathbf{q}}^*$ stand for the values to be cleaned, the projection for velocities is

$$\begin{bmatrix} \mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\mathbf{K} & \frac{\Delta t^2}{4}\Phi_{\mathbf{q}}^{\mathrm{T}} \\ \frac{\Delta t^2}{4}\Phi_{\mathbf{q}} & 0 \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}} \\ \sigma \end{Bmatrix} = \begin{Bmatrix} \left(\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\mathbf{K}\right)\dot{\mathbf{q}}^{*} \\ -\frac{\Delta t^2}{4}\Phi_t \end{Bmatrix} \tag{21}$$

and for accelerations

$$\begin{bmatrix} \mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\mathbf{K} & \frac{\Delta t^2}{4}\Phi_{\mathbf{q}}^{\mathrm{T}} \\ \frac{\Delta t^2}{4}\Phi_{\mathbf{q}} & 0 \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \kappa \end{Bmatrix} = \begin{Bmatrix} \left(\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}+\frac{\Delta t^2}{4}\mathbf{K}\right)\ddot{\mathbf{q}}^{*} \\ -\frac{\Delta t^2}{4}\left(\dot{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}+\dot{\Phi}_t\right) \end{Bmatrix} \tag{22}$$

Again, sparse matrix technology for symmetric and positive-definite matrices is used at the time of obtaining simulation results.

2.3. FULLY-RECURSIVE FORMULATION

A topological, fully-recursive algorithm of order $O(N)$ was developed by Jiménez [5] based on the *articulated inertia method* originally proposed by Featherstone [6]. To represent the state of the mechanism, a set of relatives coordinates $\mathbf{z}$ are used, as shown in Figure 1. If the kinematic chain is open, these coordinates constitute a minimum set, equal in number to the degrees of freedom, and therefore independent; else, they are dependent and so related through a certain number of constraint equations. On the other hand, each body of the mechanism is defined by the Cartesian velocities

$$\mathbf{Z}^{\mathrm{T}} = \left\{ \dot{\mathbf{r}}_0^{\mathrm{T}} \quad \omega^{\mathrm{T}} \right\} \tag{23}$$

where $\dot{\mathbf{r}}_0$ stands for the velocity of the material point of the body that is currently situated at the origin of the inertial reference frame, and ω is the angular velocity of the body.

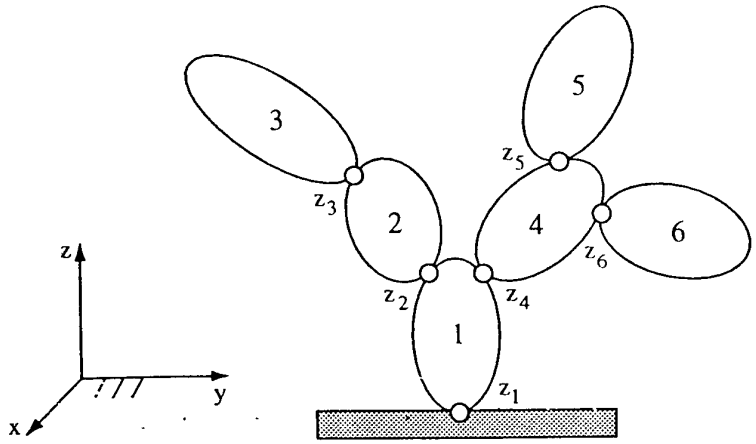


Figure 1. Configuration of a multibody system of 6 bodies and 6 relative coordinates.

Based on this kinematic assumption, the principles of dynamics lead to the following mass matrix and vector of applied forces for a certain body

$$\mathbf{M} = \begin{bmatrix} m\mathbf{I} & -m\tilde{\mathbf{r}}_G \\ m\tilde{\mathbf{r}}_G^T & \mathbf{J}_G - m\tilde{\mathbf{r}}_G\tilde{\mathbf{r}}_G \end{bmatrix} \tag{24}$$

$$\mathbf{Q} = \begin{Bmatrix} \mathbf{f} - \omega \times (\omega \times m\mathbf{r}_G) \\ \mathbf{n}_G - \omega \times \mathbf{J}_G\omega + \mathbf{r}_G \times (\mathbf{f} - \omega \times (\omega \times m\mathbf{r}_G)) \end{Bmatrix} \tag{25}$$

where m is the mass of the body, $\mathbf{r}_G$ is the 3x1 vector containing the position of the center of mass of the body, $\tilde{\mathbf{r}}_G$ is the 3x3 dual antisymmetric matrix of $\mathbf{r}_G$, $\mathbf{J}_G$ is the 3x3 inertia tensor referred to the center of mass, and, finally, $\mathbf{f}$ and $\mathbf{n}_G$ are the 3x1 vectors of applied forces and moments acting at the center of gravity of the body, respectively.

As for any recursive formulation, the accelerations are calculated, without the need of solving a set of simultaneous equations, by going through the kinematic chain in a forward and backward manner. The first step consists of a kinematic transmission starting from the base towards the end of the chain. The corresponding recursive expressions in velocities and accelerations are given by

$$\mathbf{Z}_i = \mathbf{Z}_{i-1} + \mathbf{b}_i\dot{\mathbf{z}}_i \tag{26}$$

$$\dot{\mathbf{Z}}_i = \dot{\mathbf{Z}}_{i-1} + \mathbf{b}_i\ddot{\mathbf{z}}_i + \mathbf{d}_i \tag{27}$$

where $\mathbf{b}_i$ is a 6x1 vector containing the value of the Cartesian velocities of body i, $\mathbf{Z}_i$, when all the velocities $\dot{\mathbf{z}}$ are made zero except for $\dot{\mathbf{z}}_i = 1$; and $\mathbf{d}_i$ is also a 6x1 vector containing the difference of the Cartesian accelerations $\dot{\mathbf{Z}}_i - \dot{\mathbf{Z}}_{i-1}$ when all the accelerations $\ddot{\mathbf{z}}$ are made zero. This first step remains the same whether the kinematic chain is open or closed. Obviously, the vectors $\mathbf{b}_i$ and $\mathbf{d}_i$ depend on the kind of kinematic pair that joins body *i-1* with body *i*. Equations (26) and (27) form a recursive relation between the velocities and accelerations of two consecutive bodies, in terms of the relative joint velocity and acceleration.

The second step is a condensation of the inertial and applied forces starting from the end of the chain and progressing towards the base. If the kinematic chain is open, the principle of virtual power for an *N*-link multibody system may be formulated as follows

$$\sum_{i=1}^{N} \mathbf{Z}_i^{*T}(\mathbf{M}_i\dot{\mathbf{Z}}_i - \mathbf{Q}_i) = 0 \tag{28}$$

where $\mathbf{Z}_i^*$ represents the virtual velocities of body i. Since they are not independent, they cannot be eliminated in equation (28). Introducing the recursive relations (26) and (27) in equation (28) for the body N, one may obtain:

$$\sum_{i=1}^{N-2} \mathbf{Z}_i^{*\mathrm{T}}\left(\mathbf{M}_i\dot{\mathbf{Z}}_i - \mathbf{Q}_i\right) +$$
$$\mathbf{Z}_{N-1}^{*\mathrm{T}}\left[(\mathbf{M}_{N-1}+\mathbf{M}_N)\dot{\mathbf{Z}}_{N-1} - (\mathbf{Q}_{N-1}+\mathbf{Q}_N - \mathbf{M}_N\mathbf{d}_N) + \mathbf{M}_N\mathbf{b}_N\ddot{z}_N\right] + \tag{29}$$
$$\dot{z}_N^{*\mathrm{T}}\mathbf{b}_N^{\mathrm{T}}\left[\mathbf{M}_N\mathbf{b}_N\ddot{z}_N + \mathbf{M}_N\dot{\mathbf{Z}}_{N-1} - (\mathbf{Q}_N - \mathbf{M}_N\mathbf{d}_N)\right] = 0$$

The virtual relative velocity $\dot{z}_N^*$ appears in the third line of equation (29). Since it is independent of the remaining virtual velocities, the bracket multiplying it must be zero. This condition allows us to obtain the independent accelerations:

$$\ddot{z}_N = \left(\mathbf{b}_N^{\mathrm{T}}\mathbf{M}_N\mathbf{b}_N\right)^{-1}\left[\mathbf{b}_N^{\mathrm{T}}(\mathbf{Q}_N - \mathbf{M}_N\mathbf{d}_N) - \mathbf{b}_N^{\mathrm{T}}\mathbf{M}_N\dot{\mathbf{Z}}_{N-1}\right] \tag{30}$$

Reintroducing $\ddot{z}_N$ in equation (29), and defining

$$\mathbf{K}_N = \mathbf{I}_6 - \mathbf{M}_N\mathbf{b}_N\left(\mathbf{b}_N^{\mathrm{T}}\mathbf{M}_N\mathbf{b}_N\right)^{-1}\mathbf{b}_N^{\mathrm{T}} \tag{31}$$

$$\hat{\mathbf{M}}_{N-1} = \mathbf{M}_{N-1} + \mathbf{K}_N\mathbf{M}_N \tag{32}$$

$$\hat{\mathbf{Q}}_{N-1} = \mathbf{Q}_{N-1} + \mathbf{K}_N(\mathbf{Q}_N - \mathbf{M}_N\mathbf{d}_N) \tag{33}$$

makes it possible to rewrite equation (29) in an analogous form to equation (28) but where all the terms affecting body N have been eliminated

$$\sum_{i=1}^{N-2} \mathbf{Z}_i^{*\mathrm{T}}\left(\mathbf{M}_i\dot{\mathbf{Z}}_i - \mathbf{Q}_i\right) + \mathbf{Z}_{N-1}^{*\mathrm{T}}\left(\hat{\mathbf{M}}_{N-1}\dot{\mathbf{Z}}_{N-1} - \hat{\mathbf{Q}}_{N-1}\right) = 0 \tag{34}$$

There are only N-1 terms in equation (34) and $\hat{\mathbf{M}}_{N-1}$ is the *articulated inertia* of bodies N-1 and N.

Repeating the same reasoning in a recursive manner for any relative acceleration one obtains the accelerations at any joint i:

$$\ddot{\mathbf{z}}_i = \left(\mathbf{b}_i^{\mathrm{T}}\hat{\mathbf{M}}_i\mathbf{b}_i\right)^{-1}\left[\mathbf{b}_i^{\mathrm{T}}\left(\hat{\mathbf{Q}}_i - \hat{\mathbf{M}}_i\mathbf{d}_i\right) - \mathbf{b}_i^{\mathrm{T}}\hat{\mathbf{M}}_i\dot{\mathbf{Z}}_{i-1}\right] \tag{35}$$

and the recursive condensation of applied and inertia forces which constitute the second step for open chains is performed by means of the following expressions

$$\mathbf{K}_i = \mathbf{I}_6 - \hat{\mathbf{M}}_i\mathbf{b}_i\left(\mathbf{b}_i^{\mathrm{T}}\hat{\mathbf{M}}_i\mathbf{b}_i\right)^{-1}\mathbf{b}_i^{\mathrm{T}} \tag{36}$$

$$\hat{\mathbf{M}}_{i-1} = \mathbf{M}_{i-1} + \mathbf{K}_i\hat{\mathbf{M}}_i \tag{37}$$

$$\hat{\mathbf{Q}}_{i-1} = \mathbf{Q}_{i-1} + \mathbf{K}_i\left(\hat{\mathbf{Q}}_i - \hat{\mathbf{M}}_i\mathbf{d}_i\right) \tag{38}$$

The third and last step for open chains is the calculation of the relative accelerations $\ddot{\mathbf{z}}$ from the base to the end of the chain. This calculation is carried out by recursively using equation (35).

Looking at the triple recursive procedure, it may be seen that the number of arithmetic operations grows proportionally with the number of degrees of freedom of the open chain: an $O(N)$ method.

The consideration of branches in the kinematic chain is a simple task. In the junction bodies, the forward recursive computations (steps 1 and 3) must be split into two separate procedures that move independently along each branch. In the backward recursive computations (step 2) the two separate procedures along each branch meet at the junction body and yield a single procedure.

As said before, steps 2 and 3 are not valid for closed kinematic chains. However, these two steps may be modified in order to tackle closed-loop multibody systems. To obtain it, they must be transformed into open-loop systems through the *cut-joint* method, which eliminates or *cuts* a joint in each closed-loop. This method is described in detail in [5] and [7].

In this work the method shown in [5] is followed. When a joint is removed, the corresponding constraint forces $\left(\Phi_{\mathbf{Z}}^{\mathrm{T}}\lambda\right)$ and $\left(-\Phi_{\mathbf{Z}}^{\mathrm{T}}\lambda\right)$ must be introduced on both links connected at the joint, and condensed along with the other forces. $\Phi_{\mathbf{Z}}$ is the Jacobian matrix of the constraints with respect to the body variables $\mathbf{Z}$, and λ are the Lagrange multipliers. A penalty formulation is used, so that

$$\lambda = \alpha\left(\ddot{\Phi} + 2\mu\Omega\dot{\Phi} + \Omega^2\Phi\right) \tag{39}$$

where α is the penalty factor. Equation (39) may be written as a function of the variables that define the two bodies (say r and s) connected at the cut-joint,

$$\lambda = \alpha\left(\Phi_{\mathbf{Z}}\dot{\mathbf{Z}}_r - \Phi_{\mathbf{Z}}\dot{\mathbf{Z}}_s + \gamma\right) \tag{40}$$

and

$$\gamma = \dot{\Phi}_{\mathbf{Z}}\mathbf{Z}_r - \dot{\Phi}_{\mathbf{Z}}\mathbf{Z}_s + 2\mu\Omega\left(\Phi_{\mathbf{Z}}\mathbf{Z}_r - \Phi_{\mathbf{Z}}\mathbf{Z}_s\right) + \Omega^2\Phi \tag{41}$$

Now, the second and third steps explained above for open chains must be reformulated. If the joint between bodies r and s has been removed to open a closed-loop, application of the principle of virtual power gives the following equation

$$\begin{aligned} &\sum_{\substack{i=1 \\ i\neq r,s}}^{N} \mathbf{Z}_i^{*\mathrm{T}}\left(\mathbf{M}_i\dot{\mathbf{Z}}_i - \mathbf{Q}_i\right) + \\ &\mathbf{Z}_r^{*\mathrm{T}}\left(\mathbf{M}_r\dot{\mathbf{Z}}_r - \mathbf{Q}_r + \Phi_{\mathbf{Z}}\lambda\right) + \mathbf{Z}_s^{*\mathrm{T}}\left(\mathbf{M}_s\dot{\mathbf{Z}}_s - \mathbf{Q}_s - \Phi_{\mathbf{Z}}\lambda\right) = 0 \end{aligned} \tag{42}$$

instead of equation (28) which is valid for open-loop systems. Starting from equation (42), a similar process to that developed before for open chains should be carried out. Expressions will be obtained either for mass and force condensation (second step), as for relative accelerations (third step). Nevertheless, this process requires a considerable amount of effort as it is very much problem

dependent, leading to involved expressions and tedious calculations. Consequently, general expressions like (35)-(38) cannot be provided, and they are to be developed for each particular multibody system.

Unlike the previously presented Lagrangian formulations which featured a Newton-Raphson scheme, this fully-recursive method only allows for a fixed point iteration to be used for integration purposes. The reason is that introducing the integrator equations and obtaining the tangent matrix becomes practically unfeasible. If again the trapezoidal rule is used, then positions and velocities should be expressed in terms of accelerations.

$$\mathbf{z}_{n+1} = \frac{\Delta t^2}{4}\ddot{\mathbf{z}}_{n+1} + \hat{\mathbf{z}}_n \quad \text{with} \quad \hat{\mathbf{z}}_n = \mathbf{z}_n + \Delta t\dot{\mathbf{z}}_n + \frac{\Delta t^2}{4}\ddot{\mathbf{z}}_n \tag{43}$$

$$\dot{\mathbf{z}}_{n+1} = \frac{\Delta t}{2}\ddot{\mathbf{z}}_{n+1} + \hat{\dot{\mathbf{z}}}_n \quad \text{with} \quad \hat{\dot{\mathbf{z}}}_n = \dot{\mathbf{z}}_n + \frac{\Delta t}{2}\ddot{\mathbf{z}}_n \tag{44}$$

3. The Example: Modeling Aspects

To compare the three formulations presented above, the dynamics of a complex and realistic 3D multibody system have been analyzed. The chosen example is the military 4x4 Bombardier Iltis vehicle [8], which was proposed by the European automobile industry as a benchmark problem to check multibody dynamic codes. The vehicle features four identical suspensions (see Figure 2) whose characteristics are:

– A *shock absorber*, which provides a damping force given by the following nonlinear expression:

$$F_D = 9945.627v + 33955.72v^2 - 59832.25v^3 - 395651.0v^4 \text{ [N]} \quad \text{for } -0.2<v<0.21 \text{ m/s}$$

$$F_D = -416.42 + 1844.3v \text{ [N]} \quad \text{for } v<-0.2 \text{ m/s}$$

$$F_D = 1919.1638 + 1634.727v \text{ [N]} \quad \text{for } v<0.21 \text{ m/s}$$

and a elastic force due to an external polymer, given by

$$F_S = -4.0092*10^6 + 2.8397*10^7 x - 6.7061*10^7 x^2 + 5.2796*10^7 x^3 \text{ [N]}$$

where the distance x is defined in meters.

– A *leaf spring*, modeled as a linear spring of 35,900 N/m. Moreover, a rubber bumper contacts the wheel at the leaf spring connection point after a vertical motion, from the equilibrium position of 70 mm. The bumper stiffness is assumed to be a constant value of 10^7 N/m, which is engaged only after the clearance is used up. This last element is considered in some simulations, thus adding a much higher stiffness to the problem, but not in others. Its presence will be conveniently indicated in the results below.

– A *tire*, whose radial stiffness is 460,000 N/m. The side force and aligning torque are considered by application of the tire Calspan model [9].

Also, there is an steering system, which is kinematically guided during the maneuvers. The total mass of the vehicle is about 1500 Kg.

3.1. MODELING FOR THE LAGRANGIAN FORMULATIONS

In order to perform the analysis with Lagrangian formulations, the Iltis vehicle has been modeled using fully Cartesian dependent coordinates, also called *natural* coordinates [4]. For three-dimensional multibody systems, these coordinates describe the position of the whole mechanism by means of the Cartesian coordinates of *basic points* and the Cartesian components of *unit vectors*, all distributed throughout the elements. Each body of the system should have a sufficient number of points and vectors linked to it, so that its motion is completely defined. Figure 2 illustrates how the chassis, suspension and steering system of the vehicle have been modeled.

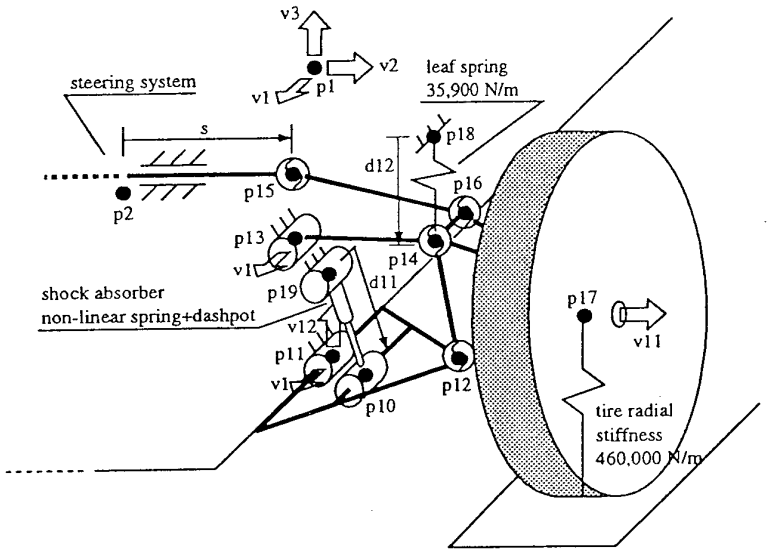


Figure 2. Modeling of the Iltis vehicle in fully Cartesian coordinates.

As may be seen in Figure 2, basic points, unit vectors and relative coordinates of distance have been used adding to a total number of 168. The system has 11 degrees of freedom: six free-body motion degrees of freedom for the chassis, one for each suspension and one more for the steering system. In consequence, the variables of the problem are related through 168-11=157 constraint equations. In practice, only 10 from the 11 degrees of freedom are true dynamic degrees of freedom because, as said before, the steering motion is kinematically guided. This guidance is introduced through a new constraint equation, so that the final number of constraints is 158.

3.2. MODELING FOR THE RECURSIVE FORMULATION

Recursive formulations require the use of relative coordinates. They define the position of each body in relation to the previous one in the kinematic chain, by using the relative degrees of freedom allowed by the joint linking those elements. Figure 3 illustrates the way the chassis, suspension and steering system of the Iltis vehicle have been modeled using this kind of coordinates.

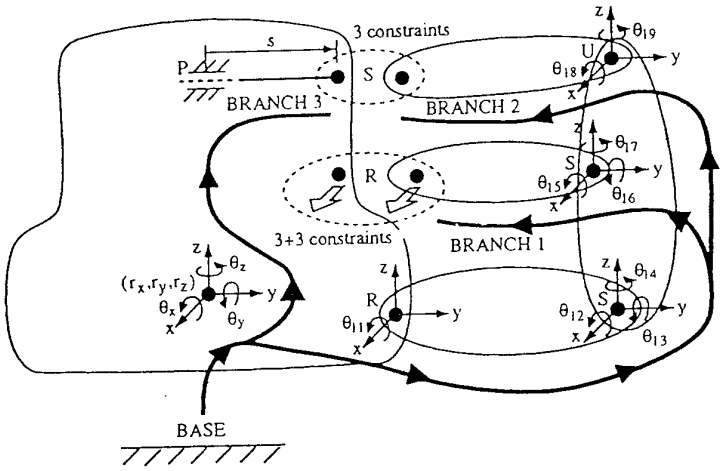


Figure 3. Modeling of the Iltis vehicle in relative coordinates.

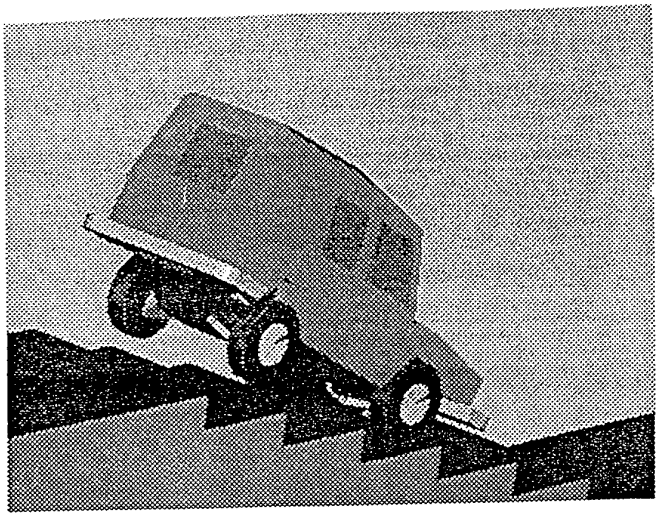
In this case, the number of variables is 43, distributed as follows: 6 for the chassis (three translations and three rotations), 9 for each suspension (all of them rotations at the different pairs as shown in Figure 3) and 1 distance for the steering system. Since this last variable is kinematically guided, the total number of variables amounts to 42. Figure 3 shows where the closed-loops have been cut. Two cuts are performed for each suspension. One of them needs 3+3=6 constraints to establish identity between points and directions at the rotational joint removed (actually, only two constraints are needed for the direction but using three is easier), while the other only introduces 3 more constraints as this time the joint removed is spherical. Therefore, the total number of constraints is 36, with just 32 independent.

4. Simulations Results

In order to test the numerical efficiency of the previously presented formulations, the Iltis vehicle is driven through three different simulations, which are described below.

4.1. SIMULATIONS DESCRIPTION

The first simulation, that will be called Simul1, consists of 10 seconds of motion with the vehicle going up an inclined ramp and the down a series of stairs at a speed of 5 m/s. The simulation leads to the strong motion with accelerations of up 5*g* shown in Figure 4.



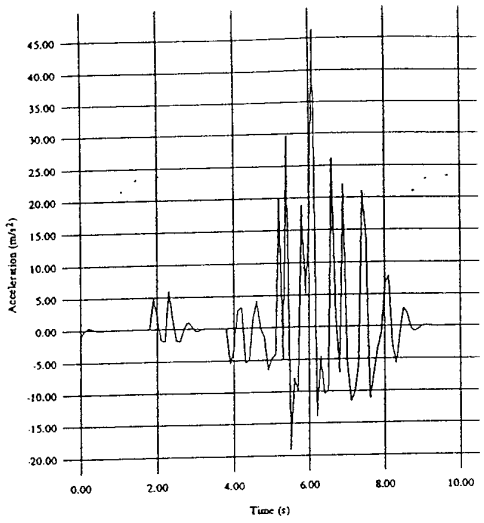
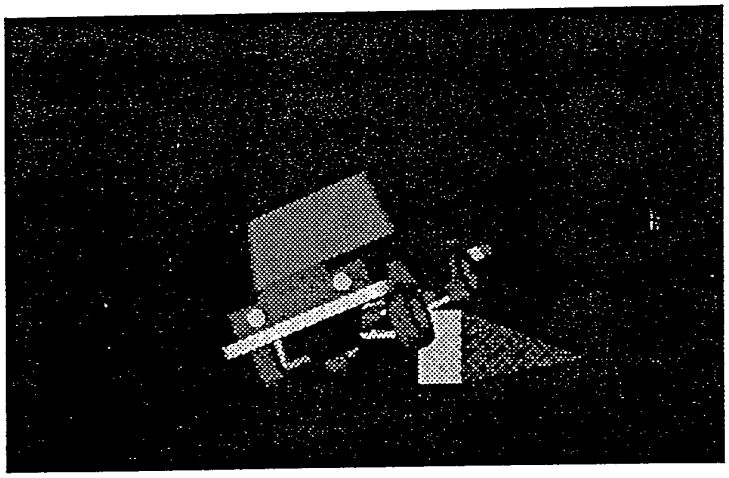


Figure 4. The Iltis vehicle performing Simul1 and vertical acceleration of its center of mass.

During the second simulation, Simul2, the vehicle goes off a ramp and hits the ground, leading to the acceleration time history of Figure 5, that also lasts for 10 seconds at a speed of 5 m/s.



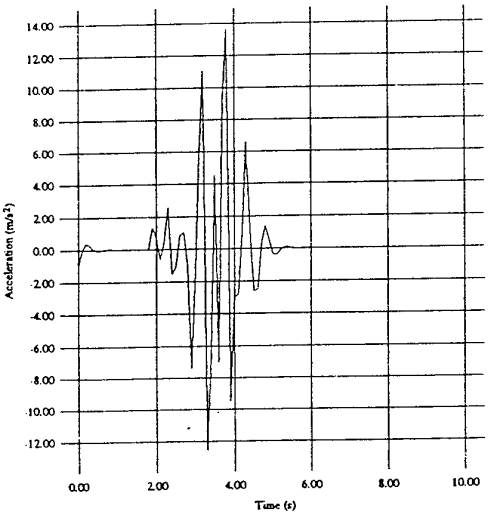


Figure 5. The Iltis vehicle performing Simul2 and vertical acceleration of its center of mass.

In the third simulation, Simul3, the Iltis vehicle is driven through a series of columns in the slalom maneuver. The simulation lasts for 10 seconds at a vehicle speed of 5 m/s, leading to the acceleration time history of Figure 6.

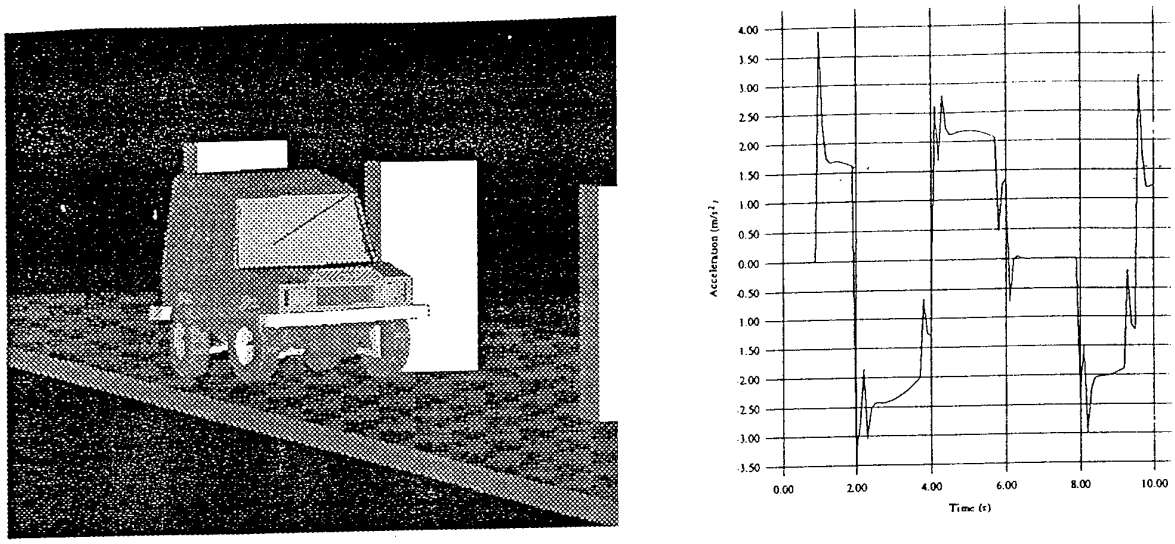


Figure 6. The Iltis vehicle performing Simul3 and transversal acceleration of its center of mass.

4.2. RESULTS ON ONE-PROCESSOR COMPUTERS

The three simulations described above were carried out on a SGI Indigo2 IMPACT with one processor R4400SC @ 200 MHz and 2 Mb of secondary cache memory, using the three formulations being compared. Table I features two columns for each formulation: the first one gives the smallest CPU time achieved with the corresponding method, while the second one shows the time step size which led to that best CPU time.

Table I illustrates the fact that the augmented Lagrangian formulation (with a penalty factor of 10^9) is clearly superior to the classical one because it performs notably faster (between three and four times) and achieves convergence for greater time step sizes. On the other hand, the fully-recursive formulation shows to be more efficient than the augmented Lagrangian method although it needs to take a smaller time step size to converge (due to the fact that it uses fixed point iteration). Thus, in simulations such as Simul2 the augmented Lagrangian method can work with larger time step sizes, and performs at the same level as the fully-recursive formulation. On the other hand, when the maneuver becomes sharper (such as in Simul1) the augmented Lagrangian method must keep on small step sizes and the fully-recursive formulation takes a fair advantage.

	Augmented Lagrangian		**Classical Lagrangian**		**Fully-Recursive**	
	CPU (s)	Δt (s)	CPU (s)	Δt (s)	CPU (s)	Δt (s)
Simul1	**34**	0.0175	**139**	0.01	**20**	0.0075
Simul2	**18**	0.03	**58**	0.025	**14**	0.0075
Simul3	**27**	0.025	**99**	0.01	**9**	0.0075

Table I. Efficiency on a single-processor machine.

4.3. RESULTS ON PARALLEL MACHINES

To find out the effect of parallelization on the three formulations under study, the simulation Simul1 has been executed on a SUN Sparc Station 20 with 4 processors HyperSparc (RT625) @ 100 MHz with 256 Kb of cache memory. Table II shows the reduction in CPU time experienced by each method as the number of processors increases.

# of processors	Augmented Lagrangian	Classical Lagrangian	Fully-Recursive
1	0.0	0.0	0.0
2	22.7	6.5	0.0
3	29.6	9.0	0.0
4	29.6	9.0	0.0

Table II. Reduction of CPU time in % due to parallelization.

It may be seen from Table II that the method that takes most advantage of the parallelization, up to 30% savings, is the augmented Lagrangian, since it uses it during the matrix formation and solution. The classical Lagrangian only takes advantage of the parallelization at the time of solving the equations, achieving up to 9% reduction in CPU time. On the other hand, the fully-recursive formulation does not profit at all from the parallel processing, thus indicating to be a method suitable for sequential processing. It must be noticed that, in all cases, the fourth processor does not contribute to improve the CPU time.

4.4 INFLUENCE OF NUMERICAL STIFFNESS

The simulation Simul1 has been carried out again using the three formulations. However, this time it has been called Simul1S, since the rubber bumpers of stiffness 10^7 N/m have been introduced into the four suspensions of the model, so that the algorithm behavior under conditions of high numerical stiffness may be observed. Table III compares the results obtained by the three methods when performing Simul1 and Simul1S.

	Augmented Lagrangian		Classical Lagrangian		Fully-Recursive	
	CPU (s)	Δt (s)	CPU (s)	Δt (s)	CPU (s)	Δt (s)
Simul1	**34**	0.0175	**139**	0.01	**20**	0.0075
Simul1S	**78**	0.005	**435**	0.0025	**34**	0.0025

Table III. Influence of numerical stiffness.

Table III shows that numerical stiffness makes all the methods reduce the time step size. The fully-recursive by a factor of 3, the augmented Lagrangian by a factor of 3.5, and the classical

Lagrangian by a factor of 4. Thus the last two lose performance with respect to the fully-recursive formulation. The penalty factor of the augmented Lagrangian procedure needs to be increased to 10^{10} to achieve convergence and, at lower time steps, as for instance 10^{-3} seconds, this formulation starts suffering from ill-conditioning effects. When numerical stiffness remains low, this problem arises at a step size of 10^{-4} or 10^{-5} seconds, but the presence of high numerical stiffness makes it appear at larger time steps.

5. Conclusions

In this paper a thorough and novel comparison of different leading methods of dynamic simulation has been carried out for large multibody systems. The conclusions that have been reached may be summarized as follows:

- The modeling process required by the fully-recursive method leads to a relative coordinate formulation that is suitable for open chain configurations, but presents serious complexities and difficulties in the case of closed kinematic chains. The modeling for the Lagrangian formulations is much simpler and may be done with dependent coordinates. Therefore the former calls for special purpose implementations whereas the latter is more suitable for general purpose simulators. In respect to modeling, an analogy may be established between the dependent coordinates and the finite element formulation for structural analysis.
- The augmented Lagrangian method presents a leading matrix that is symmetric and positive-definite, thus making it very robust to address simulations with singular configurations. Also, this method permits the differentiation of the dynamic equations to get the tangent matrix in a Newton-Raphson iteration, which allows for convergence at large time step sizes. Furthermore, it takes a serious advantage of parallel computing. However, it suffers from ill-conditioning when the time step size becomes small, thus having convergence difficulties for stiff systems. A precision environment of 32-digits could be the cure for this numerical drawback.
- The classical Lagrangian formulation generates an almost double-size problem than its augmented counterpart, and takes less profit of parallel computing, thus performing notably worse. In addition, its leading matrix is less robust, causing the failure of the integration process at singular configurations and with redundant constraints.
- The fully-recursive procedure performs well for stiff and non-stiff systems. Its involved formulation only permit a fixed-point iteration scheme, so that the time step size must be kept small to achieve convergence. Despite of this fact, it shows to be the most efficient method among the three compared, but does not obtain any advantage from parallel computing.
- According to the results presented in this paper, real-time simulation of large multibody systems can be achieved with medium-size workstations, opening a wide field for development of hardware-in-the-loop and man-in-the-loop facilities, virtual prototyping tools and intelligent vehicle control systems.

Acknowledgments

The support of this work provided by the United States Army Research Office under grant DAAH04-95-1-0662, and the CICYT of the Spanish Ministry of Education under grant TAP95-0226 is greatly acknowledged.

References

1. Cuadrado, J., Cardenal, J. and Bayo, E., 'Modeling and solution methods for efficient real-time simulation of multibody dynamics', *Multibody System Dynamics* **1**, 1997, 259-280.
2. Bayo, E. and Ledesma, R. 'Augmented Lagrangian and mass-orthogonal projection methods for constrained multibody dynamics', *Nonlinear Dynamics* **9**, 1996, 113-130.
3. Brenan, K.E, Campbell, S.L. and Petzold, L.R., *The Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, Elsevier, (1989).
4. García de Jalón, J. and Bayo, E., *Kinematic and Dynamic Simulation of Multibody Systems*. Springer-Verlag, Berlin, 1994.
5. Jiménez, J.M., 'Kinematic and dynamic formulations for real-time simulation of multibody systems', Ph.D. Thesis, University of Navarra, Spain, 1993.
6. Featherstone, R., *Robot Dynamics Algorithms*, Kluwer Academic Publishers, Dordrecht, 1987.
7. Bae, D.S., Hwang, R.S. and Haug, E.J., 'A recursive formulation for real-time dynamic simulation', *1988 Advances in Design Automation*, ed. by S.S. Rao, ASME, 1988, 499-508.
8. Iltis Data Package, IAVSD Workshop, Herbertov, Czechoslovakia, September 1990.
9. Bohn, P.F. and Keenan R.J., 'Hybrid computer vehicle handling program', Applied Physics Laboratory, The Johns Hopkins University Report no. DOT-HS-801 290, November 1974.